UNITED STATES DEPARTMENT OF THE INTERIOR

GEOLOGICAL SURVEY

POLYGON - Program Listing

by

Bruce A. Chuchel
U.S. Geological Survey
Menlo Park, CA, 94025

Open File Report 85-233-B

Prepared in cooperation with the
Nevada Operations Office
U. S. Department of Energy
(Interagency Agreement DE-AI08-78ET44802)

This report is preliminary and has not been reviewed for conformity with U.S. Geological Survey editorial standards. Use of brand names in this report is for the sake of description only, and does not constitute endorsement by the U. S. Geological Survey. Although this program has been tested, the U.S. Geological Survey makes no guarantee of correct results.

Menlo Park, California

1985

UNITED STATES
DEPARTMENT of INTERIOR
GEOLOGICAL SURVEY

POLYGON - Program Listing

BY

B. A. CHUCHEL[1]

---

[1] U.S. Geological Survey, Menlo Park, CA

INTRODUCTION

This report contains a listing of the FORTRAN program POLYGON. The Open-File Report #85-233-A [Chuchel, 1985] describes in detail how to use the POLYGON program. POLYGON features color-graphics, an interactive user dialogue, brief help messages, and "zooming" to selected portions of the terminal screen in order to edit and manipulate polygons using a graphics cursor or the Envision mouse. POLYGON has been developed in support of the U. S. Geological Survey's effort to characterize potential radioactive waste storage sites at the Nevada Test Site for the Nevada Waste Storage Investigations project.

POLYGON is a computer program for developing polygonal models on an Envision color-graphics terminal. The term "polygonal model" means (1) the coordinates of the corners of a set of polygonal shapes, (2) a set of parameters and parameter descriptions attached to each polygon, and (3) the topologic structure that links the polygons in a storage hierarchy. POLYGON uses a data structure called a quadruply linked tree [Knuth, 1969, p. 352] to store the topology of the collection of polygons composing the model.

These polygonal shapes may subsequently be used by independent modeling programs to represent three-dimensional shapes of gravitational or magnetic sources; for example, output from POLYGON is directly compatible with programs MAGPOLY or GRAVPOLY [Plouff, 1975a, 1975b; Godson 1983a, 1983b] and PFGRAV3D or PFMAG3D [Blakely, 1981].

A model may consist of up to 100 polygons; each with up to 100 vertices. Each polygon may be assigned up to ten numerical parameters, each with a descriptive label. A polygon's sides may not be self-crossing; perimeters of polygons are not allowed to cross, but any number of polygons can be completely contained within other polygons.

POLYGON is written in DEC (Digital Equipment Corporation) extended FORTRAN 77 and is presently operational on the USGS, Branch of Geophysics, VAX/VMS computer. The program is written to operate on the Envision 200 series of color-graphics terminals by Envision Technology Incorporated. The 200 series features a Tektronix 4014 compatible instruction set [Envision, 1983]. Modular construction of the program should make it relatively simple to convert to other color-graphics terminals.

REFERENCES

Blakely, R. J., 1981, A program for computing the magnetic anomaly over digital topography: U.S. Geological Survey Open-File Report 81-298, 46 p.

Chuchel, B. A., 1985, POLYGON - An interactive program for constructing and editing the geometries of polygons using a color graphics terminal: U.S. Geological Survey Open-File Report 85-233-A, 38 p.

Envision Technology, INC., 1983, Models 220 and 230 color graphics terminals reference manual #20640-1: San Jose, CA, 178 p.

Godson, R. H., 1983a, MAGPOLY: A modification of DONALD PLOUFF's 3-D magnetic modelling program: U.S. Geological Survey Open-File Report 83-345, 48p.

Godson, R. H., 1983b, GRAVPOLY: A modification of DONALD PLOUFF'S 3-D gravity modelling program: U.S. Geological Survey Open-File Report 83-346, 53p.

Knuth, D. E., 1969, The art of computer programming, Volume 1/Fundamental Algorithms: Menlo Park, CA, Addison-Wesley, 634p.

Plouff, Donald, 1975a, Derivation of formulas and FORTRAN programs to compute magnetic anomalies of prisms: U.S. Geological Survey Report, 112 p., available from National Technical Information Service, No. PB-243-525, U.S. Department of Commerce, Springfield, VA, 22161.

Plouff, Donald, 1975b, Derivation of formulas and FORTRAN programs to compute gravity anomalies of prisms: U.S. Geological Survey Report, 90 p., available from National Technical Information Service, No. PB-243-526, U.S. Department of Commerce, Springfield, VA 22161.

```
C-
C ****************************************************************************
C    Program POLYGON is designed to facilitate the construction
C    and editing of the geometries of polygons interactively
C    on an Envision color graphics terminal.
C
C    Various output files may be generated; for example, output
C    files compatible with the programs GRAVPOLY [Godson, 1983b]
C    and MAGPOLY [Godson, 1983a] may be created.  A standard
C    grid file may also be created which is compatible with the
C    programs PFGRAV3D and PFMAG3D [Blakely, 1981].
C
C    POLYGON can take input from the Envision mouse or the cursor
C    control keys on the numeric keypad.  For detailed information
C    about using the program see the USGS Open-File report #85-437-A
C    "POLYGON - An Interactive Program For Constructing and Edit-
C    ing the Geometries of Polygons Using A Color Graphics Terminal".
C
C    Author: Bruce A. Chuchel, USGS, Menlo Park, CA., 94025
C    Version 0.5   2/21/84
C    Version 1.0  12/09/84
C-
C ****************************************************************************
      Character*15 label(10)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /subscreen/xscrn(2),yscrn(2),xgrd(2),ygrd(2)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /inout/xin(100,100),yin(100,100),xout(100,100),
     &yout(100,100)
      Common /parameter /parm(100,10)
      Common /labels/label
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /grid/grd(250000)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /calc/ncont,cmin,cdel
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /names/grdnam,modnam,modgrd
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /misc/ncol,nrow,first,ntop,ifirst
      Common /flags/mcflag,votflg
      Common /model/mdflag
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
C
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character ans*1,mcflag*2,first*1,color*1
      Character votflg*1,type*1,term*1,id*56,pgm*8
      Character quest*80,com*2,grdnam*80,modnam*80,modgrd*80
C
C - Print welcoming banner.....
C
      Call wrtmsg('*****************************************************')
      Call wrtmsg('    Welcome to POLYGON version 1.0...12/09/84')
      Call wrtmsg('**********************************************************')
C
C - Get grid (grd).
C
      Call grdin(itest)
      If (itest.EQ.-1)Go To 100
      ncol=nc
      nrow=nr
C
C - Ask user if verbose or terse questioning is desired.
C
      Call askvot(votflg)
      If (votflg.EQ.'Q')Go To 100
C
C - Initialize values used in scaling and drawing grid on
C   terminal
C
      Call initial
C
C - Scale grid to screen.
C
      Call scalog2sc(0)
      xsc=float(nxpix)
      ysc=float(nypix)
      iwcorg=iwc0
      jwcorg=jwc0
      nxorgp=nxpix
      nyorgp=nypix
      Call setbnd
      Call setfil(solid)
      Call clrply
C
C - Ask for polygon command
C
      ifirst=1
      com='h'
      itest=0
      Do 10 while(itest.EQ.0)
         quest=' Polygon command (p/o/r/w/z/h/q)'
         ival=iaquest(quest,com,'(a2)',2)
C
         If (com.EQ.'P'.OR.com.EQ.'p') Then
            Call pdcom(itest2)
         Else If (com.EQ.'O'.OR.com.EQ.'o') Then
            If (mdflag.EQ.1) Then
               Call outcom(itest2)
            Else
               Call errmod
            End If
         Else If (com.EQ.'R'.OR.com.EQ.'r') Then
            Call modin(itest2)
            If (ifirst.EQ.0.AND.itest2.EQ.1) Then
               Call setclr(plyclr)
               Call setfil(open)
               Call drawwalk
               Call setfil(solid)
            End If
            If (itest2.EQ.1)mdflag=1
         Else If (com.EQ.'W'.OR.com.EQ.'w') Then
            Call modout(itest2)
         Else If (com.EQ.'Z'.OR.com.EQ.'z') Then
            Call zomcom(ifirst,itest2)
C        else if (com.eq.'S') then
C           call comcom(itest2)
```

```
          Else If (com.EQ.'H'.OR.com.EQ.'h') Then
             Call hlpcom
          Else If (com.EQ.'Q'.OR.com.EQ.'q'.OR.ival.EQ.-1) Then
             Call askend(ans)
             If (uns.EQ.'Y'.OR.ans.EQ.'y') Then
                Call modout(itest2)
             End If
             itest=1
          Else
             Call errmsg
             com='h'
          End If
          If (com.NE.'h')com='p'
   10 End Do
C
  100 Continue
      Call closup
      Stop ' End of Polygon'
      End
C ****************************************************************************
C  addply - Subroutine called by plycom that allows the user to
C           enter a new polygon and positions this polygon in the
C           topology array.
C ****************************************************************************
      Subroutine addply(itest)
      Dimension xpoly(100),ypoly(100),xloc(100),yloc(100)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /commands/nmax,epsiln,delin,delout
      Common /calc/ncont,cmin,cdel
      Common /misc/ncol,nrow,first,ntop,ifirst
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /flags/mcflag
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character lntype*1,ans*2,ans2*2,mcflag*2,first*1
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
C
      itest=1
      Call setclr(plyclr)
      lntype='0'
      Call setlin(lntype)
C
C - Ask if user needs help
C
      If (first.EQ.'y') Then
         Call hlpadd(itest)
      End If
C
C - Find the top (root) node of tree
C
      If (ntotal.GT.0) Then
         Call fndtop(ntop)
         If (ntop.LE.0) Then
            itest=-1
            Go To 100
         End If
      End If
C
C - Set scaling variables for drawing on terminal (scalen)
C
      Call scalen
```

```
C
C - Find the next available polygon npoly, if one exists.
C
   20 Continue
      ans='U'
      Call fndnum(npoly)
      If (npoly.LE.0) Then
         If (npoly.EQ.0) Then
            Call wrtmsg('  *** Polygon list full ***')
            itest=0
         Else
            Call wrtmsg('  Error in fndnum routine')
            itest=-1
         End If
         Go To 100
      End If
C
C - Print message about entry into Polygon drawing mode
C
      Call enhmsg('*** Add polygon mode ***')
C
C - Initialize the temporary array.
C
      Call inttmp
   30 Continue
C
C - Allow user to enter and draw the polygon on the terminal,
C   find the set of x,y points that define the polygon.
C
      Call getply(xloc,yloc,nbrpts,mcflag,itest)
      If (mcflag.EQ.'Q')Go To 100
      If (itest.LE.0) Then
         Call asksto(ans)
         If (nbrpts.GE.2) Then
            Call setclr(black)
            Call drwpln(xloc,yloc,nbrpts)
            Call setclr(plyclr)
         End If
      Else
         ans='U'
      End If
      If (ans.EQ.'Y')Go To 30
      If (ans.EQ.'N')Go To 100
C
C - Test polygon for being self-reentrant.
C
      Call selftest(iflag,nside1,nside2,xloc,yloc,nbrpts)
      If (iflag.LT.0) Then
         Call wrtmsg(' Error, polygon is self-crossing...try again.')
         Call setclr(black)
         Call setfil(open)
         Call drwply(xloc,yloc,nbrpts)
         Call setclr(plyclr)
         Call setfil(solid)
         Go To 30
      End If
C
      Call inttmp
      ntemp=nbrpts
      If (izoom.EQ.1) Then
         Do 50 i=1,nbrpts
         Call invers(xt,yt,xloc(i),yloc(i))
         xpoly(i)=xt
         ypoly(i)=yt
         xtemp(i)=xt
         ytemp(i)=yt
   50    Continue
      Else
         Do 60 ii=1,nbrpts
         xpoly(ii)=xloc(ii)
         ypoly(ii)=yloc(ii)
         xtemp(ii)=xloc(ii)
         ytemp(ii)=yloc(ii)
   60    Continue
      End If
C
C - Determine if the polygon npoly crosses another polygon, and
C   if it does not, place it in the tree (topology).
C
      Call fndtpl(npoly,ntop,itest)
      If (itest.EQ.1) Then
         ntotal=ntotal+1
         Call stoply(xpoly,ypoly,nbrpts,npoly,itest)
         Call fndbbx(xmin,xmax,ymin,ymax,xpoly,ypoly,nbrpts,delout)
         xminbx(npoly)=xmin
         xmaxbx(npoly)=xmax
         yminbx(npoly)=ymin
         ymaxbx(npoly)=ymax
         Call fndbpl(npoly,itest)
         Call newold
      Else
         Call setclr(black)
         Call setfil(open)
         Call drwply(xloc,yloc,nbrpts)
         Call setclr(plyclr)
         Call setfil(solid)
         Call oldnew
      End If
      If (itest.EQ.-1)Go To 100
C
C - Test ntotal here, make sure it doesn't go over.
C
      first='n'
C
C - Find the root node (ntop) of the tree
C
      Call fndtop(ntop)
C
C - Ask if another polygon will be drawn on this grid
C
      Call askply(ans2)
      If (ans2.EQ.'Y')Go To 20
      If (ans2.EQ.'N')itest=1
      If (ans2.EQ.'Q')itest=-2
C
  100 Continue
      first='n'
      Return
      End
C+
C ***************************************************************
C  addpnt - Allows the adding of a string of points into a polygon
C           between positions ncorn,ncorn2.
C-
C ***************************************************************
      Subroutine addpnt(itest)
      Dimension xline(100),yline(100)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /parameter /parm(100,10)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /flags/mcflag,votflg
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character votflg*2,lntype*1,mcflag*2,mcur*2,ans*2,ans2*2
C
      Call enhmsg('*** Add point mode ***')
C
      itest=0
      If (ntotal.GT.0) Then
C
C - If cursor type has not been selected prompt for type.
C
         If (mcflag.EQ.'N') Then
            Call askmoc(mcur)
            If (mcur.EQ.'Q') Then
               itest=-1
            Else
               mcflag=mcur
            End If
         Else
            mcur=mcflag
         End If
C
C - Print help message.
C
         If (votflg.EQ.'V')Call hlpapt
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
         Do 10 while(itest.EQ.0)
         iflag=0
         iflag2=0
         If (mcur.EQ.'M'.OR.mcur.EQ.'C') Then
C
C - Initialize the temp array.
C
            Call inttmp
C
C - Let user pick the polygon and return npoly,ncorn,x,y.
C
            Call pckply(npoly,ncorn,x,y,ans,mcur,ierr)
C
C - Ask if corner point of polygon picked is to be used.
C
            If (ans.EQ.'Y') Then
C
C - Get corner to start adding string xline,yline at.
C
               Call msgspt
               Call retpnt(xtest,ytest,mcur,ierr2)
               If (ierr2.LE.-1) Then
            itest=-1
            Go To 20
         End If
C
         If (izoom.EQ.1)Call invers(xtest,ytest,xtest,ytest)
C
C - Pick side where points to be added.
C
         Call magsp2
         Call fndsid(ncorn,ncorn2,dmin,xtest,ytest,npoly,ierr3)
         If (ierr3.LE.0) Then
            itest=-1
            Go To 20
         End If
C
C - Undraw the connecting lines between ncorn,ncorn2.
C
         x1=xscr(npoly,ncorn)
         y1=yscr(npoly,ncorn)
         x2=xscr(npoly,ncorn2)
         y2=yscr(npoly,ncorn2)
C
         If (izoom.EQ.1) Then
            Call trans(xone,yone,x1,y1)
            Call trans(xtwo,ytwo,x2,y2)
         Else
            xone=x1
            yone=y1
            xtwo=x2
            ytwo=y2
         End If
C
         Call setclr(black)
         Call drwlin(xone,yone,xtwo,ytwo)
         Call setclr(white)
C
C - Now pick which corner of the side where additions start.
C
         Call retpnt(xt,yt,mcur,itest)
         If (itest.LE.-1)Go To 20
         If (izoom.EQ.1)Call invers(xt,yt,xt,yt)
C
C - Determine orientation of line by finding closest point to xt,yt.
C
         dist1=(xt-x1)**2+(yt-y1)**2
         dist2=(xt-x2)**2+(yt-y2)**2
         If (dist1.LT.dist2.AND.dist1.LT.delin**2) Then
            xfirst=x1
            yfirst=y1
            xend=x2
            yend=y2
            iset=1
         Else If (dist2.LT.dist1.AND.dist2.LT.delin**2) Then
            xfirst=x2
            yfirst=y2
            xend=x1
            yend=y1
            iset=0
         Else
            iset=-1
         End If
         If (iset.EQ.-1)Go To 20
C
C - Translate the coordinates xfirst,yfirst,xend,yend into zoomed
```

```
C   coordinates if izoom=1
C
               If (izoom.EQ.1) Then
                  Call trans(xfirst,yfirst,xfirst,yfirst)
                  Call trans(xend,yend,xend,yend)
               End If
C
C - User enters string of points (xline,yline) to be added.
C
               nbrpts=numply(npoly)
               nline=0
               idraw=0
               lntype='0'
               Call setlin(lntypo)
               Call getlin(xline,yline,nline,xfirst,yfirst,xend,yend,
     &         idraw,nbrpts,nptmax,mcflag,itest)
               If (itest.LE.-1) Then
                  iflag2=1
                  Go To 20
               End If
C
C - Convert line to unzoomed screen coordinate if izoom=1.
C
               If (izoom.EQ.1) Then
                  Do 50 kk=1,nline
                  Call invers(xline(kk),yline(kk),xline(kk),
     &            yline(kk))
   50             Continue
               End If
               istrg=nbrpts-ncorn
C
C - Construct the xtemp,ytemp arrays.
C
               Do 60 j=1,ncorn
               xtemp(j)=xscr(npoly,j)
               ytemp(j)=yscr(npoly,j)
   60          Continue
C
C - Now bring xline,yline into xtemp,ytemp
C
               If (iset.EQ.1) Then
                  ncon=0
                  ipar=1
               Else
                  ncon=nline+1
                  ipar=-1
               End If
C
               Do 70 jj=1,nline
               xtemp(ncorn+jj)=xline(ncon+ipar*jj)
               ytemp(ncorn+jj)=yline(ncon+ipar*jj)
   70          Continue
C
C - Now add the points in xscr,yscr from ncorn2 to nbrpts onto
C   xtemp,ytemp at position ncorn+nline+1.
C
               If (istrg.GT.0) Then
                  Do 80 k=1,istrg
                  xtemp(ncorn+nline+k)=xscr(npoly,ncorn+k)
                  ytemp(ncorn+nline+k)=yscr(npoly,ncorn+k)
   80             Continue
               End If
               ntemp=nbrpts+nline
C
```

```
C - Test the new polygon for self-crossing.
C
               Call selftest(iflag3,nside1,nside2,xtemp,ytemp,ntemp)
               If (iflag3.EQ.-1) Then
                  Call wrtmsg(' Error, polygon in self-crossing')
                  iflag2=1
                  Go To 20
               End If
C
C - Test the new polygon in the xtemp,ytemp array and fit
C   into the topology structure.
C
               Call testopo(npoly,itest2)
               If (itest2.GE.0)Then
                  iflag=1
               else
                  iflag2=1
               end if
               itest=1
C
   20          Continue
               If (iflag.EQ.0) Then
                  Call sutclr(plyclr)
                  Call setfil(open)
                  Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
                  If (iflag2.eq.1) Then
                     If (iset.EQ.1) Then
                        ilow=1
                        iup=nline
                     Else
                        ilow=nline
                        iup=1
                     End If
                     Call setclr(black)
                     Call drwlin(xone,yone,xline(ilow),yline(ilow))
                     Call drwpln(xline,yline,nline)
                     Call drwlin(xline(iup),yline(iup),xtwo,ytwo)
                     Call setclr(plyclr)
                  End If
               End If
            Else If (ans.EQ.'N') Then
               itest=0
            Else
               itest=-1
            End If
            Call setfil(solid)
            If (ierr.EQ.-1)itest=-1
         Else If (mcur.EQ.'Q') Then
            itest=-1
         End If
   10 End Do
      Else
         Call wrtmsg(' Sorry, no polygons')
         itest=-1
      End If
C
      Return
      End
C+
C ****************************************************************
C  alphon - Turns the alphanumeric cursor back on.
C-
C ****************************************************************
      Subroutine alphon
```

```
C
      Call esccom('a2')
C
      Return
      End
C+
C **************************************************************
C  askall - Asks if user would like to view/change the parameters
C           associated with the enhanced polygon.
C-
C **************************************************************
      Subroutine askall(ans)
      Character quest*80,ans*2
C
      ans='y'
      quest=' View/change parameters of enhanced (white) polygon'
      Call askynq(quest,ans,-1,2)
C
      Return
      End
C+
C **************************************************************
C  askans - Asks the user to confirm that a previous action is
C           to be carried out.
C-
C **************************************************************
      Subroutine askans(ans)
      Character ans*2,quest*80
C
      ans='n'
      quest=' Do you wish to proceed'
      Call askynq(quest,ans,0,2)
C
      Return
      End
C+
C **************************************************************
C  askclr - Asks for information used to draw a Denver standard
C           grid in color on the Envision terminal.
C
C           ncont = Number of contour intervals
C                 = 0 = Sets up terminal for polygon drawing only
C                 = 1-12 = Draws grid with the number of contours
C                          indicated
C                 = // = Quit
C
C           cmin  = Minimum contour level
C
C           cdel  = Contour interval
C-
C **************************************************************
      Subroutine askclr(ncont,cmin,cdel,itest)
      Character quest*80
C
      cmin=0.0
      cdel=0.0
C
      Write (6,15)
   15 Format (/,' To draw the grid in color I first need some',
     &' parameters.',
     &/,' Enter the number of contours',
     &/,'            0 = Sets up terminal for polygon drawing only',
     &/,'         1-12 = Draws grid with number of contours indicated',
     &/,'           // = Quit',/)
```

```
C
C - Get the number of contours to use for coloring grid.
C
      itest=0
      Do 10 while(itest.EQ.0)
         ncont=0
         quest=' *Number of contours (0-12, // to quit)'
         ival=iiquest(quest,ncont,'(i2)',0)
C
         If (ival.EQ.-1) Then
            itest=-1
         Else If (ncont.GE.0.AND.ncont.LE.12) Then
            itest=1
         Else
            Call errmsg
         End If
   10 End Do
      If (itest.EQ.-1.OR.ncont.EQ.0)Go To 100
C
C - Get minimum contour level.
C
      quest=' *Give minimum contour level'
      cmin=0.0
      ival=irquest(quest,cmin,'(e16.8)',0)
      itest=ival
      If (itest.EQ.-1)Go To 100
C
C - Get contour interval.
C
      quest=' *Give contour interval'
      cdel=0.0
      ival=irquest(quest,cdel,'(e16.8)',0)
      itest=ival
      If (itest.EQ.0)itest=1
C
  100 Continue
      Return
      End
C+
C **************************************************************
C  askdvl - Asks for the dval to use in initializing a grid.
C-
C **************************************************************
      Subroutine askdvl(dval,itest)
      Character quest*80
      dval='ffff7fff'x
C
      itest=1
   10 Continue
      quest=' Dval for grid'
      ival=irquest(quest,dval,'(e16.8)',16)
      If (ival.EQ.-1)itest=-1
C
      Return
      End
C+
C **************************************************************
C  askend - ASK END - Asks the user before quitting if the
C           model should be written to an output file.
C-
C **************************************************************
      Subroutine askend(ans)
      Character ans*2,quest*80
C
```

```
      ans='y'
      quest=' Save model'
      Call askynq(quest,ans,1,1)
C
      Return
      End
C+
C ************************************************************
C  askenh - Asks user if polygon drawn in white is correct.
C-
C ************************************************************
      Subroutine askenh(ans)
      Character ans*2,quest*80
C
      ans='Y'
      quest='  Is the polygon drawn in white the correct one'
      Call askynq(quest,ans,-1,2)
C
      Return
      End
C+
C ************************************************************
C  askgvl - Asks the user when generating a GRAVPOLY model file
C           which parameters in the parameter list should be
C           assigned to:
C
C           ione   = Height of top of body
C           itwo   = Height of bottom of body
C           ithree = Density contrast of body
C
C-
C ************************************************************
      Subroutine askgvl(ione,itwo,ithree,itest)
      Common /labels/label
      Character ans*2,quest*80,label(10)*15
      Character use1*15,use2*15,use3*15
C
    5 Continue
      ione=0
      itwo=0
      ithree=0
      use1='UNASSIGNED'
      use2='UNASSIGNED'
      use3='UNASSIGNED'
C
C - Message about parameters that program needs.
C
      Call msggvl
      Call wait
C
C - Display the current labels.
C
      Call msglab
C
C - Ask for position in parameter list of height of top of body.
C
      itest=0
      Do 10 while(itest.EQ.0)
         ione=0
         quest=' Top of body (1-10, 0 or // to quit)'
         ival=iiquest(quest,ione,'(I2)',0)
C
         If (ione.GE.1.AND.ione.LE.10) Then
            use1=label(ione)
            itest=1
         Else If (ione.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   10 End Do
      If (itest.EQ.-1)Go To 50
C
C - Ask for position in parameter list of height of bottom of body.
C
      itest=0
      Do 20 while(itest.EQ.0)
         itwo=0
         quest=' Bottom of body (1-10, 0 or // to quit)'
         ival=iiquest(quest,itwo,'(i2)',0)
C
         If (itwo.GE.1.AND.itwo.LE.10) Then
            use2=label(itwo)
            itest=1
         Else If (itwo.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   20 End Do
      If (itest.EQ.-1)Go To 50
C
C - Ask for position in parameter list of density contrast.
C
      itest=0
      Do 30 while(itest.EQ.0)
         ithree=0
         quest=' Density contrast of body (1-10, 0 or // to quit)'
         ival=iiquest(quest,ithree,'(i2)',0)
C
         If (ithree.GE.1.AND.ithree.LE.10) Then
            use3=label(ithree)
            itest=1
         Else If (ithree.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   30 End Do
      If (itest.EQ.-1)Go To 50
C
C - Display the labels that the user selected and prompt if these
C   are correct, allow changing if wrong.
C
      itest=0
      Do 40 while(itest.EQ.0)
         Print *,' You made the following assignments:'
         Print *,'    Height of top of body is labeled: ',use1
         Print *,'    Height of bottom of body is labeled: ',use2
         Print *,'    Density contrast of body is labeled: ',use3
         ans='Y'
         quest=' Are these assignments correct (y/n/q)'
         ival=iaquest(quest,ans,'(a2)',-2)
C
         If (ans.EQ.'Y') Then
            itest=1
         Else If (ans.EQ.'N') Then
            itest=2
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   40 End Do
      If (itest.EQ.2)Go To 5
C
   50 Continue
      If (itest.EQ.-1) Then
         ione=0
         itwo=0
         ithree=0
      End If
C
      Return
      End
C+
C **************************************************************
C  asklab - ASK LABel - Asks the user for the label name to be
C           associated with a value array.
C-
C **************************************************************
      Subroutine asklab(label,itest)
      Character labnum*2,oldlab*15,label*15,quest*80
C
      If (itest.GE.1.AND.itest.LE.10) Then
         If (itest.EQ.1) labnum='1'
         If (itest.EQ.2) labnum='2'
         If (itest.EQ.3) labnum='3'
         If (itest.EQ.4) labnum='4'
         If (itest.EQ.5) labnum='5'
         If (itest.EQ.6) labnum='6'
         If (itest.EQ.7) labnum='7'
         If (itest.EQ.8) labnum='8'
         If (itest.EQ.9) labnum='9'
         If (itest.EQ.10) labnum='10'
         oldlab=label
         itest=0
         Do 10 while(itest.EQ.0)
            quest=' Enter label # '//labnum
            ival=iaquest(quest,label,'(a15)',15)
C
            If (label.EQ.'//'.OR.ival.EQ.-1) Then
               label='UNASSIGNED'
               itest=-1
            Else If (label.NE.' ') Then
               itest=1
            Else
               label=oldlab
               Call errmsg
            End If
   10    End Do
C
      Else
         Call wrtmsg(' Error in asklab, itest out of range')
         itest=-1
      End If
c
      Return
      End
C+
C **************************************************************
C  askmdl - ASK MoDel - Asks for the name of the user's model.
```

```
C-
C ****************************************************************
      Subroutine askmdl(modnam,itest)
      Character tempnm*80,modnam*80,quest*80
C
      tempnm=modnam
      itest=0
      Do 10 while(itest.EQ.0)
         quest=' Model name'
         ival=iaquest(quest,modnam,'(a80)',80)
C
         If (ival.EQ.-1) Then
            itest=-1
         Else If (modnam.NE.' ') Then
            itest=1
         Else If (modnam.EQ.' ') Then
            modnam=tempnm
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C ****************************************************************
C  askmoc - ASK Mouse Or Cursor - Asks the user to select the
C           Envision mouse or cursor keys for entry of locations
C
C           mcur = Mouse/Cursor
C                  (note: answer is returned upshifted)
C                  m = Mouse (default)
C                  c = Cursor keys
C                  q = Quit
C
C-
C ****************************************************************
      Subroutine askmoc(mcur)
      Character mcur*2,quest*80
C
      itest=0
      If (itest.EQ.0) Then
10       Continue
         mcur='m'
         quest=' Mouse or cursor keys (m/c/q)'
         ival=iaquest(quest,mcur,'(a2)',-2)
C
         If (ival.EQ.-1.OR.mcur.EQ.'Q') Then
            mcur='Q'
            itest=-1
         Else If (mcur.EQ.'M'.OR.mcur.EQ.'C') Then
            itest=1
         Else
            Call errmsg
         End If
         If (itest.EQ.0) Go To 10
      End If
C
      Return
      End
C+
C ****************************************************************
C  askmpp - Asks the user what method to user when picking a
C           polygon.
C-
```

```
C *************************************************************
      Subroutine askmpp(ans)
      Character ans*2,quest*80
C
      itest=0
      ans='h'
      Do 10 while(itest.EQ.0)
         quest=' Method for picking polygon (c/m/h/q)'
         ival=iaquest(quest,ans,'(a2)',-2)
C
         If (ans.EQ.'C'.OR.ans.EQ.'M') Then
            itest=1
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            ans='Q'
            itest=1
         Else If (ans.EQ.'H') Then
            Call hlpmpp
         Else
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C *************************************************************
C  askmvl - Asks the user when generating a MAGPOLY model file
C           which parameters in the parameter list should be assigned to:
C
C           ione   = Height of top of body
C           itwo   = Height of bottom of body
C           ithree = Volume magnetic susceptibility (emuX1000000)
C           ifour  = Remanent or total volume magnetization
C                    (emuX1000000).
C           ifive  = Declination of remanent or total magnetization
C                    in degrees, measured positive clockwise from
C                    the direction of the y axis.
C           isix   = Inclination of remanent or total magnetization
C                    in degrees, measured positive downward from
C                    the horizontal plane.
C
C
C-
C *************************************************************
      Subroutine askmvl(ione,itwo,ithree,ifour,ifive,isix,itest)
      Common /labels/label
      Character ans2*2,ans*2,quest*80,label(10)*15
      Character*15 use1,use2,use3,use4,use5,use6
C
    5 Continue
      ione=0
      itwo=0
      ithree=0
      ifour=0
      ifive=0
      isix=0
      use1='UNASSIGNED'
      use2='UNASSIGNED'
      use3='UNASSIGNED'
      use4='UNASSIGNED'
      use5='UNASSIGNED'
      use6='UNASSIGNED'
C
C - Message about what information is to be provided.
C
```

```
      Call msgmvl
      Call wait
C
C - Display the current labels.
C
      Call msglab
C
C - Ask for position in parameter list of height of top of body.
C
      itest=0
      Do 10 While (itest.EQ.0)
         ione=0
         quest=' Top of body (1-10, 0 or // to quit)'
         ival=iiquest(quest,ione,'(i2)',0)
C
         If (ione.GE.1.AND.ione.LE.10) Then
            use1=label(ione)
            itest=1
         Else If (ione.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   10 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for position in parameter list of height of bottom of body.
C
      itest=0
      Do 20 While (itest.EQ.0)
         itwo=0
         quest=' Bottom of body (1-10, 0 or // to quit)'
         ival=iiquest(quest,itwo,'(i2)',0)
C
         If (itwo.GE.1.AND.itwo.LE.10) Then
            use2=label(itwo)
            itest=1
         Else If (itwo.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   20 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for position in parameter list of Volume magnetic susceptibility.
C
      itest=0
      Do 30 While (itest.EQ.0)
         ithree=0
         quest=' Volume magnetic susceptibility (1-10, 0 or // to quit)'
         ival=iiquest(quest,ithree,'(i2)',0)
C
         If (ithree.GE.1.AND.ithree.LE.10) Then
            use3=label(ithree)
            itest=1
         Else If (ithree.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   30 End Do
      If (itest.EQ.-1)Go To 100
C
```

```
C - Ask for position in parameter list of remanent or total volume
C   magnetization.
C
      itest=0
      Do 40 While (itest.EQ.0)
          ifour=0
          quest=' Remanent or total volume magnetization (1-10,
     & 0 or // to quit)'
          ival=iiquest(quest,ifour,'(i2)',0)
C
          If (ifour.GE.1.AND.ifour.LE.10) Then
             use4=label(ifour)
             itest=1
          Else If (ifour.EQ.0.OR.ival.EQ.-1) Then
             itest=-1
          Else
             Call errmsg
          End If
   40 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for position in parameter list of Declination of remanent or
C   total magnetization.
C
      itest=0
      Do 50 While (itest.EQ.0)
          ifive=0
          quest=' Declination of remanent or total magnetization
     & (1-10, 0 or // to quit)'
          ival=iiquest(quest,ifive,'(i2)',0)
C
          If (ifive.GE.1.AND.ifive.LE.10) Then
             use5=label(ifive)
             itest=1
          Else If (ifive.EQ.0.OR.ival.EQ.-1) Then
             itest=-1
          Else
             Call errmsg
          End If
   50 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for position in parameter list of Inclination of remanent or
C   total magnetization.
C
      itest=0
      Do 60 While (itest.EQ.0)
          isix=0
          quest=' Inclination of remanent or total magnetization
     & (1-10, 0 or // to quit)'
          ival=iiquest(quest,isix,'(i2)',0)
C
          If (isix.GE.1.AND.isix.LE.10) Then
             use6=label(isix)
             itest=1
          Else If (isix.EQ.0.OR.ival.EQ.-1) Then
             itest=-1
          Else
             Call errmsg
          End If
   60 End Do
      If (itest.EQ.-1)Go To 100
C
C - Display the labels that the user selected and prompt if these
```

```
C   are correct, allow changing if wrong.
C
C65    continue
      Itest=0
      Do 70 While (Itest.EQ.0)
         Write (6,60)
   60 Format (/,' You have made the following assignments:',/)
         Print *,'     Height of top of body is labeled: ',use1
         Print *,'     Height of bottom of body is labeled: ',use2
         Print *,'     Volume magnetic susceptibility is labeled: ',use3
         Print *,'     Remanent or total volume magnetization is
     &labeled:',use4
         Print *,'     Declination of remanent or total magnetization
     &is labeled:',use5
         Print *,'     Inclination of remanent or total magnetization
     &is labeled:',use6
         ans2='Y'
         quest=' Are these assignments correct (y/n/q)'
         ival=iaquest(quest,ans2,'(a2)',-2)
C
         If (ans2.EQ.'Y') Then
C           nlabel(2,1)=ione
C           nlabel(2,2)=itwo
C           nlabel(2,3)=ithree
C           nlabel(2,4)=ifour
C           nlabel(2,5)=ifive
C           nlabel(2,6)=isix
C           lval='M'
C           call setlbl(lval)
            itest=1
         Else If (ans2.EQ.'N') Then
            itest=2
         Else If (ans2.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   70 End Do
      If (itest.EQ.2)Go To 5
C
  100 Continue
      If (itest.EQ.-1) Then
         ione=0
         itwo=0
         ithree=0
         ifour=0
         ifive=0
         isix=0
      End If
C
      Return
      End
C+
C ************************************************************
C  asknam - ASK NAMe - Asks the question in string quest for
C          an 80 character string (name).  Formats the answer
C          using Saltus' iaquest subroutine.
C-
C ************************************************************
      Subroutine asknam(name,quest,itest)
      Character name*80,quest*80
C
      itest=0
      Do 10 while(itest.EQ.0)
```

```
          name=' '
          ival=iaquest(quest,name,'(a80)',0)
C
          If (ival.EQ.-1.OR.name.EQ.'//') Then
             itest=-1
          Else If (name.NE.' ') Then
             itest=1
          Else
             Call errmsg
          End If
   10 End Do
C
      Return
   .  End
C+
C **************************************************************
C  askok  - Asks users if current state information displayed at
C           terminal is OK.
C-
C **************************************************************
      Subroutine askok(ans)
      Character quest*80,ans*2
C
      ans='Y'
      quest=' * OK'
      Call askynq(quest,ans,-1,2)
C
      Return
      End
C+
C **************************************************************
C  askply - Asks if user wants to draw another polygon.
C-
C **************************************************************
      Subroutine askply(ans)
      Character ans*2,quest*80
C
      ans='y'
      quest=' Draw another polygon'
      Call askynq(quest,ans,-1,2)
C
      Return
      End
C+
C *************************************************************
C  askpnt - Asks user if corner of polygon selected is to be used.
C-
C *************************************************************
      Subroutine askpnt(ans)
      Character ans*2,quest*80
C
      ans='Y'
      quest=' Use the corner where the polygon was selected'
      Call askynq(quest,ans,-1,2)
C
      Return
      End
C+
C *************************************************************
C  askqst - Asks the question in the string quest.
C-
C *************************************************************
      Subroutine askqst(ans,quest,itest)
      Character ans*2,quest*80
```

```
C
      itest=0
      Do 10 while(itest.EQ.0)
         ans=' '
         ival=iaquest(quest,ans,'(a2)',0)
C
         If (ival.EQ.-1.OR.ans.EQ.'//') Then
            itest=-1
         Else If (ans.NE.' ') Then
            itest=1
         Else
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C ***************************************************************
C  askrot - Asks the user if the corner of the polygon picked
C           should be used to rotate the polygon about.
C-
C ***************************************************************
      Subroutine askrot(ans)
      Character quest*80,ans*2
C
      ans='Y'
      quest='  Rotate polygon about corner picked'
      Call askynq(quest,ans,-1,2)
C
      Return
      End
C+
C ***************************************************************
C  askrt2 - Asks the user if another corner of the polygon should
C           be used to rotate about instead.
C-
C ***************************************************************
      Subroutine askrt2(ans)
      Character quest*80,ans*2
C
      ans='Y'
      quest='  Rotate polygon about a different corner of this polygon'
      Call askynq(quest,ans,-1,2)
C
      Return
      End
C+
C ***************************************************************
C  ASKSTO - ASKS IF USER WANTS TO START OVER DRAWING POLYGON
C           CALLED BY SUBROUTINE ADDPLY (ADD POLYGON).
C-
C ***************************************************************
      Subroutine asksto(ans)
      Character quest*80,ans*2
C
      ans='y'
      quest=' Start over entering polygon'
      Call askynq(quest,ans,-1,1)
C
      Return
      End
C+
C ***************************************************************
```

```
C  asksub - Asks the user for the locations (ncmin,ncmax,nrmin,
C           nrmax) of the subgrid.
C-
C ***************************************************************
      Subroutine asksub(itest)
      Dimension xgrid(2),ygrid(2),xscrn(2),yscrn(2)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Character id*56,pgm*8,lntype*1
C
      itest=0
      Do 15 while(itest.EQ.0)
         Write (6,20)
   20 Format (/,' Enter ncmin,ncmax,nrmin,nrmax (0''s to',
     &' quit): ',$)
         Read (5,25)ncmin,ncmax,nrmin,nrmax
   25 Format (4I3)
C
         If ((ncmin.EQ.0.0).AND.(ncmax.EQ.0.0).AND.(nrmin.EQ.0.0).
     &AND.(nrmax.EQ.0.0)) Then
            itest=-1
         Else If ((ncmin.LE.0).OR.(ncmax.LE.0).OR.
     &(nrmin.LE.0).OR.(nrmax.LE.0)) Then
            Write (6,30)
   30    Format (/,' Illegal value entered...try again')
         Else If ((ncmin.GT.nc).OR.(ncmax.GT.nc).OR.(ncmin.GE.ncmax))
     & Then
            Write (6,35)
   35    Format (/,' Error in column boundaries...try again')
C
         Else If ((nrmin.GT.nr).OR.(nrmax.GT.nr).OR.(nrmin.GE.nrmax))
     & Then
            Write (6,40)
   40    Format (/,' Error in row boundaries...try again')
         Else
            itest=1
C
            xgrid(1)=ncmin
            xgrid(2)=ncmax
            ygrid(1)=nrmin
            ygrid(2)=nrmax
C
            Call fndscn(xscrn,yscrn,xgrid,ygrid,2)
C
            lntype='6'
            Call setlin(lntype)
            Call drwlin(xscrn(1),yscrn(1),xscrn(1),yscrn(2))
            Call drwlin(xscrn(1),yscrn(2),xscrn(2),yscrn(2))
            Call drwlin(xscrn(2),yscrn(2),xscrn(2),yscrn(1))
            Call drwlin(xscrn(2),yscrn(1),xscrn(1),yscrn(1))
C
            Call zomstr(itest2)
C
         End If
   15 End Do
C
      Return
      End
C+
C ***************************************************************
C  asktwz - Prints twilight zone message if user wants to enter
C           polygon with only two points.
C-
C ***************************************************************
```

```
      Subroutine asktwz(ans)
      Character quest*80,ans*2
C
      Write (6,10)
   10 Format (/,' Are you in the twilight zone???  You have only',
     &' entered two points...')
C
      itest=0
      If (itest.EQ.0) Then
   20    Continue
         ans='e'
         quest=' Do you wish to enter a point or quit (e/q)'
         ival=iaquest(quest,ans,'(a2)',0)
C
         If (ival.EQ.-1) Then
            ans='Q'
            itest=1
         Else If (ans.EQ.'E'.OR.ans.EQ.'Q') Then
            itest=1
         Else
            Call errmsg
         End If
         If (itest.EQ.0)Go To 20
      End If
C
      Return
      End
C+
C ****************************************************************
C  asktyp - Asks for the type boundary picking to be employed in
C           the zoom command.
C
C           itype = Flag for type of boundaries
C                 = 1 = Cursor/Mouse entered screen boundaries
C                 = 2 = Subgrid boundaries given as ncmin,ncmax,
C                       nrmin,nrmax
C-
C ****************************************************************
      Subroutine asktyp(itype,itest)
      Character quest*80
C
      itest=0
      Call msgsub
      Do 10 while(itest.EQ.0)
         itype=1
         quest=' Which method for boundaries (1/2)'
         ival=iiquest(quest,itype,'(i2)',2)
C
         If (ival.EQ.-1.OR.itype.EQ.0) Then
            itest=-1
         Else If (itype.NE.1.AND.itype.NE.2) Then
            Call errmsg
         Else
            itest=1
         End If
   10 End Do
C
      Return
      End
C+
C ****************************************************************
C  askval - ASK VALue - Asks for which one of the ten parameters in
C           in the parm* arrays to use when resetting the grid.
C-
```

```
C **************************************************************
      Subroutine askval(iset,ulabel,itest)
      Common /labels/label
      Character ulabel*15,label(10)*15,quest*80
C
C - Print message about selecting value (msgval).
C
      Call msgval
C
C - Display the current labels.
C
      Call msglab
C
C - Ask question, test answer (iset), and assign label if
C   answer is valid.
C
      itest=0
      Do 10 while(itest.EQ.0)
         quest=' Which parameter to use (1-10, 0 or // to quit)'
         ival=iiquest(quest,iset,'(i2)',0)
C
         If (iset.GE.1.AND.iset.LE.10) Then
            ulabel=label(iset)
            itest=1
         Else If (iset.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C **************************************************************
C  askvot - Asks user if verbose ("V") or terse ("T") prompts
C           is desired.
C-
C **************************************************************
      Subroutine askvot(ans)
      Character ans*2,quest*80
C
      itest=0
      Do 10 while(itest.EQ.0)
         ans='t'
         quest=' Do you want verbose or terse prompts (v/t/q)'
         ival=iaquest(quest,ans,'(a2)',-2)
C
         If (ival.EQ.-1) Then
            ans='Q'
            itest=1
         Else If (ans.EQ.'V'.OR.ans.EQ.'T'.OR.ans.EQ.'Q') Then
            itest=1
         Else
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C **************************************************************
C  askynq - Constructs a yes/no/quit question from the text string
C           passed to it under control of the parameters IDEF, and
```

```
C           IYNQ.  Answer is returned in character*2.
C
C           IDEF = Flag for if default answer is allowed.
C                = 1 - Default is allowed, answer returned as entered
C                = 0 - no default is allowed, and upshift answer.
C                =-1 - Upshift answer and allow default.
C
C           IYNQ = Flag for type of Yes/No/Quit prompt
C                = 2 - (y/n/q) is added to text string, user supplied
C                      default is tested and used.
C                = 1 - (y/n) is added to text string, user supplied
C                      default is tested and used.
C                =-1 - (y/n) is added to text string, "n" is used
C                      as the default answer.
C                =-2 - (y/n/q) is added to text string, "q" is used
C                      as the default answer.
C
C
C-
C ****************************************************************
      Subroutine askynq(quest,ans,idef,iynq)
      Character prompt*88,quest*80,ans*2,defans*2
C
      iflag=iabs(iynq)
      If (iflag.EQ.1.OR.iflag.EQ.2) Then
         lenqst=itlen(quest)
         If (iflag.EQ.1) Then
            prompt=quest(1:lenqst)//' (y/n)'
         Else
            prompt=quest(1:lenqst)//' (y/n/q)'
         End If
C
         If (idef.NE.0) Then
            If (iflag.EQ.2) Then
               If (iynq.EQ.-2) Then
                  ans='q'
               Else
                  If ((ans.NE.'y'.AND.ans.NE.'Y').AND.
     &            (ans.NE.'n'.AND.ans.NE.'N').AND.
     &            (ans.NE.'q'.AND.ans.NE.'Q'))ans='q'
               End If
            Else
               If (iynq.EQ.-1) Then
                  ans='n'
               Else
                  If ((ans.NE.'y'.AND.ans.NE.'Y').AND.
     &            (ans.NE.'n'.AND.ans.NE.'N'))ans='n'
               End If
            End If
         End If
C
         iset=2*idef
C
         If (idef.GE.-1.AND.idef.LE.1) Then
            itest=0
            defans=ans
   10       Continue
            ival=iaquest(prompt,ans,'(a2)',iset)
C
            If (ival.EQ.-1) Then
               If (iflag.EQ.2)ans='Q'
               If (iflag.EQ.1)ans='N'
               itest=1
            Else If ((ans.EQ.'Y'.OR.ans.EQ.'y').OR.
     &      (ans.EQ.'N'.OR.ans.EQ.'n').OR.
     &      ((ans.EQ.'Q'.OR.ans.EQ.'q').AND.iflag.EQ.2)) Then
               itest=1
            Else
               ans=defans
               Call errmsg
            End If
            If (itest.EQ.0)Go To 10
         End If
      End If
C
      Return
      End
C*
C **************************************************************
C  boundpoly - Find an inner and out bouding polygon around
C              the polygon passed in the arrays xpoly,ypoly.
C
C-
C **************************************************************
      Subroutine boundpoly(itest,xin,yin,xout,yout,xpoly,ypoly,nbrpts,
     & delin,delout)
      Dimension xpoly(nbrpts),ypoly(nbrpts),xin(nbrpts),yin(nbrpts),
     & xout(nbrpts),yout(nbrpts)
C
      If (nbrpts.LE.2) Then
         itest=-1
      Else
C
         Do 10 i=1,nbrpts
         If (i.EQ.1) Then
            xlast=xpoly(nbrpts)
            ylast=ypoly(nbrpts)
            xnext=xpoly(2)
            ynext=ypoly(2)
         Else If (i.EQ.nbrpts) Then
            xlast=xpoly(nbrpts-1)
            ylast=ypoly(nbrpts-1)
            xnext=xpoly(1)
            ynext=ypoly(1)
         Else
            xlast=xpoly(i-1)
            ylast=ypoly(i-1)
            xnext=xpoly(i+1)
            ynext=ypoly(i+1)
         End If
C
         xavec=xlast-xpoly(i)
         yavec=ylast-ypoly(i)
C
         xbvec=xnext-xpoly(i)
         ybvec=ynext-ypoly(i)
C
C - Find the angle theta between vectors A and B (xavec,yavec,
C   ...
C        i) Find the vector Dot product of A*B
C       ii) Find the norm of the vectors A, B.
C
         dotprd=xavec*xbvec+yavec*ybvec
         anorm=sqrt(xavec*xavec+yavec*yavec)
         bnorm=sqrt(xbvec*xbvec+ybvec*ybvec)
         if (anorm.lt.1.0e-12) anorm=1.0e-12
         if (bnorm.lt.1.0e-12) bnorm=1.0e-12
C
         theta=acos(dotprd/(anorm*bnorm))
         angle=theta/2.0
C
         xdvec=cos(angle)*xavec-sin(angle)*yavec
         ydvec=sin(angle)*xavec+cos(angle)*yavec
C
         dnorm=sqrt(xdvec*xdvec+ydvec*ydvec)
         if (dnorm.lt.1.0e-12) dnorm=1.0e-12
C
         xdunit=xdvec/dnorm
         ydunit=ydvec/dnorm
C
         xtest=xpoly(i)+xdunit
         ytest=ypoly(i)+ydunit
C
C - Test the two points (xtest,ytest) to see if they are within
C   or outside of the polygon npoly.  This determines how to
C   find the coordinates of the polygon bounded on the inside by
C   delin and outside by delout.
C
         Call plytst(xpoly,ypoly,nbrpts,xtest,ytest,inout)
C
         If (inout.EQ.1) Then
            parity=+1
         Else
            parity=-1
         End If
C
         hypin=abs(delin/sin(angle))
         hypout=abs(delout/sin(angle))
C
C - Now construct the corner points for the inner and outer bouinding
C   polygon.
C
         xin(i)=xpoly(i)+parity*hypin*xdunit
         yin(i)=ypoly(i)+parity*hypin*ydunit
         xout(i)=xpoly(i)-parity*hypout*xdunit
         yout(i)=ypoly(i)-parity*hypout*ydunit
   10    Continue
         itest=1
      End If
C
      Return
      End
C+
C ***********************************************************
C  chgall - Allows the changing of all values associated with
C           every polygon in the tree.  Starts at top (root)
C           node of tree, enhances polygon, prompts for change,
C           allows setting of values and finds next node (poly-
C           gon) in tree.
C-
C ***********************************************************
      Subroutine chgall
      Dimension xpoly(100),ypoly(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character ans*2,open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
C
```

```
      Call fndtop(ntop)
      If (ntop.GE.1) Then
C
C - Print message about assigning values to polygon in white.
C
         Call msgall
         Call wait
C
         ngon=ntop
         next=ntop
         itest=0
         Do 10 while(itest.EQ.0)
         npoly=info(ngon)
C
C - Enhance polygon and prompt
C
         Call setclr(white)
         Call setfil(open)
         nbrpts=numply(npoly)
         If (izoom.EQ.1) Then
            Do 20 i=1,nbrpts
            Call trans(xpoly(i),ypoly(i),xscr(npoly,i),yscr(npoly,i))
   20       Continue
         Else
            Do 25 j=1,nbrpts
            xpoly(j)=xscr(npoly,j)
            ypoly(j)=yscr(npoly,j)
   25       Continue
         End If
         Call drwclp(xpoly,ypoly,nbrpts,xleft,xright,ybot,ytop)
C
C - Ask if user wants to see/set values
C
         Call askall(ans)
C
         If (ans.EQ.'Y') Then
            iset=0
            Call valchg(npoly,iset,ierror)
         Else If (ans.EQ.'Q') Then
            itest=-1
         End If
C
C - Unenhance polygon
C
         Call setclr(plyclr)
         Call setfil(open)
         Call drwclp(xpoly,ypoly,nbrpts,xleft,xright,ybot,ytop)
         Call setfil(solid)
C
         If (itest.EQ.0) Then
            Call walk(next,ngon,ierr2)
            If (ierr2.EQ.1) Then
               ngon=next
            Else If (ierr2.EQ.0) Then
               itest=1
            Else If (ierr2.LE.-1.OR.next.LT.0) Then
               itest=-1
            End If
         End If
   10    End Do
      Else
         Call wrtmsg('  Sorry, no polygons')
      End If
C
```

```
      Return
      End
C+
C **************************************************************
C  chglab - CHanGe LABel - Allows changing the labels associ-
C           ated with the value* arrays.
C-
C **************************************************************
      Subroutine chglab(itest)
      Common /labels/label
      Character quest*80,label(10)*15,ulabel*15
C
C - Display the current settings for the labels.
C
      Call msglab
C
C - Ask which label to change, test answer, and set label.
C
      itest=0
      Do 10 while(itest.EQ.0)
      ilab=-1
      quest=' Label to change (1-10, -1 for all, 0 or // when done)'
      ival=iiquest(quest,ilab,'(I2)',2)
C
      If (ival.EQ.-1.OR.ilab.EQ.0) Then
         itest=1
      Else If (ilab.EQ.-1) Then
         i=1
         itest=0
         Do 25 while(itest.EQ.0)
         ulabel=label(i)
         ierror=1
         Call asklab(ulabel,ierror)
         If (ierror.EQ.1) Then
            label(i)=ulabel
            i=i+1
         Else If (ierror.EQ.-1) Then
            itest=-1
         End If
         If (i.GT.10)itest=1
   25    End Do
      Else If (ilab.GE.1.AND.ilab.LE.10) Then
         ulabel=label(ilab)
         ierror=ilab
         Call asklab(ulabel,ierror)
         If (ierror.EQ.1) Then
            label(ilab)=ulabel
         Else If (ierror.EQ.-1) Then
            itest=-1
         End If
      Else
         Call errmsg
      End If
   10 End Do
C
      Return
      End
C+
C **************************************************************
C  chgpar - Allows the changing of parameters associated with polygons
C-
C **************************************************************
      Subroutine chgpar(itest)
      Character quest*80,ans*2
```

```
C
      Call enhmsg('*** Change parameters ***')
C
      itest=0
      ans='h'
      Do 10 while(itest.EQ.0)
         quest=' Change parameter mode (a/p/h/q)'
         ival=iaquest(quest,ans,'(a2)',-2)
C
         If (ans.EQ.'A') Then
            Call chgall
         Else If (ans.EQ.'P') Then
            Call chgpck
         Else If (ans.EQ.'H') Then
            Call hlppar
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=1
         Else
            Call errmsg
         End If
         ans='q'
   10 End Do
C
      Return
      End
C+
C **************************************************************
C  chgpck - Allows the parameters for individual polygons to be
C           changed.
C-
C **************************************************************
      Subroutine chgpck
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /flags/mcflag,votflg
     &nrmaxz(5)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character votflg*2,mcflag*2,mcur*2,ans*2,ans2*2
C
      itest=0
C
C - Test the number of polygons ntotal, exit if <=0.
C
      If (ntotal.GT.0) Then
C
C - If cursor type has not been selected prompt for type.
C
         If (mcflag.EQ.'N') Then
            Call askmoc(mcur)
            If (mcur.EQ.'Q') Then
               itest=-1
            Else
               mcflag=mcur
            End If
         Else
            mcur=mcflag
         End If
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
```

```
         Do 10 while(itest.EQ.0)
         If (mcur.EQ.'M'.OR.mcur.EQ.'C') Then
C
C - Find polygon, enhance and return npoly,ncorn,x,y.
C
            Call pckply(npoly,ncorn,x,y,ans,mcur,ierr)
            If (ierr.LE.-1) Then
               itest=-1
               Go To 10
            End If
C
C - When correct polygon is found, prompt for parameters.
C
C
            If (ans.EQ.'Y') Then
               iset=0
               Call valchg(npoly,iset,itest2)
               itest=1
            Else If (ans.EQ.'N') Then
               itest=0
            Else
               itest=-1
            End If
C
            Call setclr(plyclr)
            Call setfil(open)
            Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
            Call setfil(solid)
         Else If (mcur.EQ.'Q') Then
            itest=-1
         End If
   10    End Do
      Else
         Call wrtmsg(' Sorry, no polygons')
         itest=-1
      End If
C
      Return
      End
C+
C *************************************************************
C  chgply - CHanGe PoLYgon - Driver for Change polygon para-
C           meter mode.  Options are:
C
C              l = Labels
C              p = Parameters associated with polygons
C              h = Help
C              q = Quit and return to Polygon add/.../edit mode
C
C-
C **************************************************************
      Subroutine chgply(itest)
      Character quest*80,ans*1
C
      Call enhmsg('*** Change_parm mode ***')
C
      itest=0
      ans='h'
      Do 10 while(itest.EQ.0)
         quest=' Change polygon parameters (l/p/h/q)'
         ival=iaquest(quest,ans,'(a2)',2)
C
         If (ans.EQ.'L'.OR.ans.EQ.'l') Then
            Call chglab(ierror)
         Else If (ans.EQ.'P'.OR.ans.EQ.'p') Then
            Call chgpar(ierror)
         Else If (ans.EQ.'H'.OR.ans.EQ.'h') Then
            Call hlpchg
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1.OR.ans.EQ.'q') Then
            itest=1
         Else
            Call errmsg
         End If
         ans='q'
   10 End Do
C
      Return
      End
C+
C ***************************************************************
C  clipper - Clips the line segment (x1,y1),(x2,y2) passed to it
C            to fit in the window defined by the x,y:
C
C                xleft<x<xright
C                ybot<y<ytop
C
C          Clipper assumes that the line segment passed to it does
C          indeed cross the given window; use tstend to determine
C          whether a line segment does indeed cross window.
C-
C ***************************************************************
      Subroutine clipper(x,y,x1,y1,x2,y2,xleft,xright,ybot,ytop)
C
      delx=x2-x1
      dely=y2-y1
C
      If (abs(dely).LT.1.e-16)dely=sign(1.e-16,dely)
      If (abs(delx).LT.1.e-16)delx=sign(1.e-16,delx)
C
      slope=dely/delx
      b=y2-slope*x2
C
C - Clip along top or bottom edges.
C
      x=x1
      If (y1.GT.ytop) Then
         x=(ytop-b)/slope
         y=ytop
      Else If (y1.LT.ybot) Then
         x=(ybot-b)/slope
         y=ybot
      End If
c
c - Clip along left or right edges.
c
      If (x.GT.xright) Then
         x=xright
         y=slope*xright+b
      Else If (x.LT.xleft) Then
         x=xleft
         y=slope*xleft+b
      End If
C
      Return
      End
C+
C ********************************************************************
C closup - Closes up graphics operations on the Envision.
```

```
C-
C *********************************************************************
      Subroutine closup
C
      Call graphoff
C
      Return
      End
C+
C **************************************************************
C  clrply - Sets up the values used for coloring a polygon.
C-
C **************************************************************
      Subroutine clrply
      Common /colors/plyclr,black,white
      Common /calc/ncont,cmin,cdel
      Character plyclr*1,black*1,white*1
C
      black='0'
      white='1'
      If (ncont.GT.0) Then
         plyclr=char(ncont+3+ichar('0'))
      Else If (ncont.EQ.0) Then
         plyclr=char(ncont+4+ichar('0'))
      End If
C
      Return
      End
C+
C **********************************************************************
C  clrsgd - Subroutine to color a grid on the Envision screen.
C
C           Author: Robert W. Simpson, USGS, Menlo Park, CA., 11/83.
C
C           Converted to a subroutine by Bruce Chuchel, USGS,
C              Menlo Park, CA., 2/84.
C
C-
C **********************************************************************
      Subroutine clrsgd(ifirst,itest)
      Common /gridspecs/id,pgm,ncol,nrow,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /scale/ xsc,ysc,xstart,ystart,xinit,yinit
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /cmdstring/ cmdstring
      Common /cmdlength/lengstr
      Common /iodevice/iounit
      Common /grid/grd(250000)
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /calc/ncont,cmin,cdel
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character plyclr*1,black*1,white*1,open*1,solid*1,filtyp*1
      Character id*56,pgm*8,rect*13,wcbp*5,ans*2,esc*1
      Character color*3,actwind*1,cmdstring*200
      Parameter (esc=char(27))
C
      lengstr=0
C
C - Find range of values
C
      If (ifirst.EQ.0)Go To 30
      Call gmaxmin
   20 Continue
      Call askclr(ncont,cmin,cdel,itest)
      If (itest.EQ.-1)Go To 100
   30 Continue
      actwind='0'
      iounit=6
C
C - Scale grid to screen
C
      If (ifirst.EQ.1) Then
         Call scaleg2sc(1)
      Else
         Call scaleg2sc(0)
      End If
      If (ifirst.EQ.1.AND.ncont.GT.0) Then
         Call askok(ans)
      End If
      If (ans.EQ.'N'.AND.ifirst.EQ.1)Go To 20
      If (ans.EQ.'Q') Then
         itest=-1
         Go To 100
      End If
C
C - Set up color screen and spectrum
C
      Call binsetup2(iounit,actwind)
      ncolors=ncont+1
      nshift=0
      If (ifirst.EQ.1) Then
         Call envclr(iounit,ncolors,nshift)
         Call inkjet(iounit,ncolors)
      End If
C
C - Set full screen to lowest color
C
      iclr=0
      Call sendcmd(iounit,color(iclr))
      If (ncont.EQ.0)Go To 50
C
C - Go thru grid once for each color...
C
      Do 50 ic=1,ncolors
      iclr=ic
      Call sendcmd(iounit,color(iclr))
      Do 50 j=nrmin,nrmax
      ileft=ncmax+1
      irt=ncmin-2
      Do 50 i=ncmin,ncmax
C - Determine color from grid value...
      If (grd(i+(j-1)*ncol).GE.1.0e+38) Then
         iclr=0
C    this sets dvals to black
      Else
         iclr=nint((grd(i+(j-1)*ncol)-cmin)/cdel+0.5)+1
         iclr=max(iclr,1)
         iclr=min(iclr,ncolors)
      End If
C - make rectangles from contiguous boxed of the same color
C    in the present row...
      If (iclr.GE.ic) Then
         ileft=min(ileft,i)
         irt=max(irt,i)
         If (irt.EQ.ncmax)Call sendcmd(iounit,rect(ileft,j,irt,j))
      Else If (irt.EQ.i-1) Then
         Call sendcmd(iounit,rect(ileft,j,irt,j))
         ileft=ncmax+1
      End If
   50 Continue
      If (ncont.EQ.0) Then
         xl=xinit
         yb=yinit
         xr=xinit+(ncmax-ncmin+1)*nxpix
         yt=yinit+(nrmax-nrmin+1)*nypix
         Call setclr(white)
         Call setfil(open)
         Call drwbox(xl,yb,xr,yt)
      End If
C - Finish up...
      Call sendcmd(iounit,'end')
  100 Continue
C
      Return
      End
C+
C **********************************************************************
C - Prints Maximum, Minimum, Average and Standard Deviation
C   about the grid file.
C-
C **************************************************************
      Subroutine gmaxmin
      Common /grid/grd(250000)
      Common /gridspecs/id,pgm,nc,nr,nx,xo,dx,yo,dy
      Character id*56,pgm*8
      ngood=0
      zmax=grd(1)
      zmin=grd(1)
      zsum=0.
      zsqsum=0.
      Do 45 j=1,nr
      Do 45 i=1,nc
      zval=grd(i+(j-1)*nc)
      If (zval.LT.1.0e38) Then
         ngood=ngood+1
         zmin=amin1(zmin,zval)
         zmax=amax1(zmax,zval)
         zsum=zsum+zval
         zsqsum=zsqsum+zval**2
      End If
   45 Continue
      zave=zsum/ngood
      zsdev=sqrt((zsqsum-zave**2)/ngood)
      twosdev=2*zsdev
      Print *,' '
      Print *,' Here are some facts about the grid values.'
      Print *,' '
      Print *,'   MIN, MAX, AVE=',zmin,zmax,zave
      Print *,'   PLUS/MIN 2 SD=',zave-twosdev,zave+twosdev
C
      Return
      End
C+
C **********************************************************************
C  binsetup2 - Sets up the Envision terminal (window actwind)
C              for graphics.
C-
C **************************************************************
      Subroutine binsetup2(iounit,actwind)
      Character esc*1,actwind*1
      Parameter (esc=char(27))
C - Erase screen...
```

```
      Write (iounit,1001)esc//'[2J'
C - Move cursor home...
      Write (iounit,1001)esc//'[H'
C - Set scrolling to line 1 to 15...
C       write(iounit,1001) esc//'[1;15r'
C - Set binary number mode...
      Write (iounit,1001)esc//'OR0'
C - Set active window number...
      Write (iounit,1001)esc//'OA'//actwind
C - Erase graphics screen to current background color...
      Write (iounit,1001)esc//'F'
C - Kill active window...
      Write (iounit,1001)esc//'OK'
C - Set figure fill...(?)
      Write (iounit,1001)esc//'CF'
 1001 Format (1h$,a)
C
      Return
      End
C **********************************************************************
      Character*13 Function box(x,y,dx,dy)
C - Draws a box dx wide and dy high centered about
C     real world coordinate point x,y.
      Character esc*1,wcbp*5
      Parameter (esc=char(27))
      Call rc2wc(x-dx/2,y-dy/2,ill,jll)
      Call rc2wc(x+dx/2,y+dy/2,iur,jur)
C - ur=upper right; ll=lower left...
      box=esc//'OX'//wcbp(ill,jll)//wcbp(iur,jur)
      Return
      End
C+
C **********************************************************************
C - Draws a rectangle which includes the boxes about
C     grid points from igll,jgll (lower left), to
C     igur,jgur (upper right)...
C     Note that the two grid points can be the same point,
C     or point in the same col or row...
C-
C **************************************************************
      Character*13 Function rect(igll,jgll,igur,jgur)
      Character erc*1,wcbp*5,id*56,pgm*8
      Common /gridspecs/id,pgm,ncol,nrow,nz,xo,dx,yo,dy,iproj,cm,bl
      Parameter (esc=char(27))
      Call corners(igll,jgll,iwc1ll,jwc1ll,iwc1ur,jwc1ur)
      Call corners(igur,jgur,iwc2ll,jwc2ll,iwc2ur,jwc2ur)
C
C - Box 1 is the ll box, box 2 is the ur...
      rect=esc//'OX'//wcbp(iwc1ll,jwc1ll)//wcbp(iwc2ur,jwc2ur)
C
      Return
      End
C **********************************************************************
      Subroutine rc2wc(x,y,iwc,jwc)
C      Converts real coordinates to world (pixel) coords
      Character id*56,pgm*8
      Common /gridspecs/id,pgm,ncol,nrow,nz,xo,dx,yo,dy
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
C     Change real world coordinates to grid coords...
      xi=(x-xo)/dx
      yj=(y-yo)/dy
C     Change grid coordinates to world (pixel) coordinates...
      iwc=iwc0+nint(real(nxpix)*xi)
      jwc=jwc0+nint(real(nypix)*yj)
      Return
      End
C **********************************************************************
      Subroutine corners(ig,jg,ill,jll,iur,jur)
C       Finds corners of the box which is centered at grid point (ig,jg).
C      Coords of the corners are returned in world (pixel) coordinates...
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      iwc=iwc0+(ig-ncmin)*nxpix
      jwc=jwc0+(jg-nrmin)*nypix
      ill=iwc-nxpix/2
      jll=jwc-nypix/2
      iur=ill+nxpix-1
      jur=jll+nypix-1
      Return
      End
C **********************************************************************
      Subroutine sendcmd(iounit,cmd)
      Character cmd*(*),cmdstring*200
      Common /cmdlength/lengstr
      Common /cmdstring/cmdstring
      If (cmd(1:3).NE.'end') Then
         leng=len(cmd)
         cmdstring(lengstr+1:lengstr+leng)=cmd(1:leng)
         lengstr=lengstr+leng
      End If
      If (lengstr.GE.70.OR.cmd(1:3).EQ.'end') Then
         Write (iounit,'(1x,a)')cmdstring(1:lengstr)
         lengstr=0
      End If
      Return
      End
C+
C **************************************************************
C     Returns envision command string to set a color.
C     Positive iclr values refer to the color scale set by subroutine
C       setcolors...
C        iclr =0 is background color = black
C        iclr =1 thru 13 refers to color spectrum set in sub setclrs6
C                 for colors 3 thru 15.
C-
C **************************************************************
      Character*3 Function color(iclr)
      Character esc*1
      Parameter (esc=char(27))
C
      If (iclr.EQ.0) Then
         color=esc//'C'//char(ichar('0'))
      Else
         color=esc//'C'//char(ichar('0')+iclr+2)
      End If
C
      Return
      End
C+
C **************************************************************
C  clspnt - Returns the corner number (NCORN) and distance (DMIN)
C           of the corner in array xpoly,ypoly closest to the
C           point (x,y).  A delta radius around each point can
C           be used for selecting or rejecting the corner.
C
C
C         ncorn - Corner number of polygon.
C                > 0 - Corner number of polygon was found (within
C                      radius used)
C                 = 0 - Corner number was not found (within radius
C                      used)
C                 =-1 - Error, the number of points of the polygon
C                      passed to clspnt was less than 1.
C
C            icon - Flag controlling which radius criterion to use
C                 = 1 = Use DELTA provided by call to subroutine
C                 = 0 = Use the maximum machine value VAXMAX available
C
C-
C ***************************************************************
      Subroutine clspnt(ncorn,dmin,x,y,xpoly,ypoly,nbrpts,delta,icon)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
      Parameter (vaxmin=-1.7e+38,vaxmax=1.7e+38)
C
      If (nbrpts.GE.1) Then
         ncorn=0
         If (icon.EQ.1) Then
            radius=delta**2
         Else
            radius=vaxmax
         End If
         dmin=vaxmax
C
         Do 10 i=1,nbrpts
         d=(x-xpoly(i))**2+(y-ypoly(i))**2
         If (d.LT.radius.AND.d.LT.dmin) Then
            ncorn=i
            dmin=d
         End If
   10    Continue
      Else
         ncorn=-1
      End If
C
      Return
      End
C+
C *****************************************************************
C  copyspecs - Copies the gridspecs to gridspecs2 common block.
C-
C *****************************************************************
      Subroutine copyspecs
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /gridspecs2/id2,pgm2,nc2,nr2,nz2,xo2,dx2,yo2,dy2,
     &iproj2,cm2,bl2
      Character id*56,id2*56,pgm*8,pgm2*8
C
      nc2=nc
      nr2=nr
      nz2=nz
      xo2=xo
      dx2=dx
      yo2=yo
      dy2=dy
      iproj2=iproj
      cm2=cm
      bl2=bl
C
      Return
      End
C+
C ***************************************************************
```

```
C cpchars - Draws a string of character-precicison characters (STRG)
C           at the world coordinates (X,Y).
C
C           Author: Robert W. Simpson
C
C-
C **************************************************************
      Subroutine cpchars(x,y,string)
      Character wcbp*5,ncbp*3,string*(*),cmd*90
C
      ix=jnint(x)
      iy=jnint(y)
      Call rdeblank(string,string,leng)
      cmd='OC'//wcbp(ix,iy)//ncbp(leng)//string(1:leng)
      Call esccom(cmd(1:10+leng))
C
      Return
      End
C+
C **************************************************************
C  cpcharset - Sets up the character-precision character parameters
C              for the Envision terminal.  See page 7-21 in Envision
C              reference manual for an explanation of the arguments.
C
C              Author: Robert W. Simpson
C
C-
C **************************************************************
      Subroutine cpcharset(ix,iy,zoom,rot,slant)
      Character*1 zoom,rot,slant
      Character wcbp*5,cmd*11
C
      cmd='Or'//wcbp(ix,iy)//zoom//rot//slant//'0'
      Call esccom(cmd)
C
      Return
      End
C+
C *************************************************************
C  cpyply - Allows the copying of a polygon from one position on
C           the screen to another position.
C-
C *************************************************************
      Subroutine cpyply(itest)
      Dimension vstore(10),xply(100),yply(100)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /inout/xin(100,100),yin(100,100),xout(100,100),
     &yout(100,100)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /parameter /parm(100,10)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /flags/mcflag,votflg
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /gridspecs/ id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /scale/ xsc,ysc,xstart,ystart,xinit,yinit
      Common /original/ iwcorg,jwcorg,nxorgp,nyorgp
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character id*56,pgm*8,votflg*2,mcflag*2,mcur*2,ans*2,ans2*2
C
      Call anhmsg('*** Copy polygon mode ***')
C
      itest=0
C
C - Test the number of polygons ntotal, exit if <=0.
C
      If (ntotal.GT.0) Then
C
C - If cursor type has not been selected prompt for type.
C
         If (mcflag.EQ.'N') Then
            Call askmoc(mcur)
            If (mcur.EQ.'Q') Then
               itest=-1
            Else
               mcflag=mcur
            End If
         Else
            mcur=mcflag
         End If
C
C - Print help message
C
         If (votflg.EQ.'V')Call hlpcpy
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
         Do 10 while(itest.EQ.0)
         iflag=0
         If (mcur.EQ.'M'.OR.mcur.EQ.'C') Then
C
C - Initialize the temp arrays.
C
            Call inttmp
C
C - Let user pick polygon and return npoly,ncorn,x,y.
C
            Call pckply(npoly,ncorn,x,y,ans,mcur,ierr)
C
C - Ask if corner point of polygon picked should be used.
C
            If (ans.EQ.'Y') Then
               nbrpts=numply(npoly)
               Call pckpnt(ncorn,dist,x,y,ans2,npoly,mcur)
               If (ans2.EQ.'Q') Then
                  itest=-1
                  Go To 20
               End If
C
C - Message about repositioning cursor to new location
C
               Call hlpmv2
               Call retpnt(xscnew,yscnew,mcur,ierr)
               If (ierr.LE.-1) Then
                  itest=-1
                  Go To 20
               End If
C
```

```
               If (izoom.EQ.1) Then
                  Call invers(xnew,ynew,xscnew,yscnew)
               Else
                  xnew=xscnew
                  ynew=yscnew
               End If
C
C - Find the offset to add to the polygon coordinates (delx,dely)
C
               delx=xnew-x
               dely=ynew-y
C
C - Store new polygon
C
               ntemp=nbrpts
               Do 30 i=1,ntemp
               xtemp(i)=xscr(npoly,i)+delx
               ytemp(i)=yscr(npoly,i)+dely
   30          Continue
C
C - Test to see if polygon will be off of unzoomed grid.
C
               xlfunz=xinit
               xrgunz=xinit+nc*nxorgp
               ybtunz=yinit
               ytpunz=yinit+nr*nyorgp
               Call testoff(xtemp,ytemp,ntemp,xlfunz,xrgunz,ybtunz,
     & ytpunz,intotal)
               If (intotal.LE.0) Then
                  If (intotal.EQ.0) Call wrtmsg(' Error, polygon will
     & be off of unzoomed grid')
                  Go To 20
               End If
C
C - Find available polygon position
C
               Call fndnum(npoly2)
C
C - Test polygon in xtemp,ytemp.
C
               Call fndtop(ntop)
               Call fndtpl(npoly2,ntop,itest3)
C
               If (itest3.EQ.1) Then
                  ntotal=ntotal+1
                  Call stoply(xtemp,ytemp,ntemp,npoly2,ierr)
                  Call fndbbx(xmin,xmax,ymin,ymax,xtemp,ytemp,
     &            ntemp,delout)
                  xminbx(npoly2)=xmin
                  xmaxbx(npoly2)=xmax
                  yminbx(npoly2)=ymin
                  ymaxbx(npoly2)=ymax
C
C - Copy the inner and outer bounding polygons offset by the
C   appropriate amount.
C
                  Do 90 l=1,ntemp
                  xin(npoly2,l)=xin(npoly,l)+delx
                  yin(npoly2,l)=yin(npoly,l)+dely
                  xout(npoly2,l)=xout(npoly,l)+delx
                  yout(npoly2,l)=yout(npoly,l)+dely
   90             Continue
C
C - Copy the parameter information
```

```
C
                  Do 95 ll=1,10
                  parm(npoly2,ll)=parm(npoly,ll)
   95             Continue
                  If (izoom.EQ.1) Then
                     Do 80 ii=1,ntemp
                     Call trans(xtemp(ii),ytemp(ii),
     &               xtemp(ii),ytemp(ii))
   80                Continue
                  End If
                  Call setclr(plyclr)
                  Call setfil(open)
                  Call drwclp(xtemp,ytemp,ntemp,xleft,xright,ybot,ytop)
                  Call setfil(solid)
                  Call newold
               Else
                  Call oldnew
               End If
               itest=1
C
   20          Continue
               If (iflag.EQ.0) Then
                  Call setclr(plyclr)
                  Call setfil(open)
                  Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
               End If
            Else If (ans.EQ.'N') Then
               itest=0
            Else
               itest=-1
            End If
            Call setfil(solid)
            If (ierr.EQ.-1)itest=-1
         Else If (mcur.EQ.'Q') Then
            itest=-1
         End If
   10    End Do
      Else
         Call wrtmsg('  Sorry, no polygons')
         itest=-1
      End If
C
      Return
      End
C+
C *******************************************************************
C  crtgrd - Prompts the user for information to construct a Denver
C           standard grid.  (Note: nc*nr<=250000).
C-
C *******************************************************************
      Subroutine crtgrd(itest)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Character quest*80,id*56,pgm*8,ans*2
C
      id=' '
      pgm='POLYGON '
      nc=0
      nr=0
      nz=1
      xo=0.0
      dx=0.0
      yo=0.0
      dy=0.0
      iproj=0
      cm=0.0
      bl=0.0
C
      Write (6,10)
   10 Format (/,' You will now be asked to enter the grid specificat',
     &'ions. ',/,' (Note: ncol*nrow less than or equal to 250,000)',/,
     &' enter // to quit.',/)
C
   15 Continue
      quest='    Enter ID for grid'
      ival=iaquest(quest,id,'(a56)',0)
      If (id.EQ.' ')Go To 15
      If (ival.EQ.-1)Go To 100
C
C - Enter the number of columns (ncol)
C
      itest=0
      Do 20 while(itest.EQ.0.OR.(nc*nr).GT.250000)
   12    Continue
         quest='    NCOL (// to quit)'
         ival=iiquest(quest,nc,'(i5)',0)
         If (nc.LE.0)Go To 12
         If (ival.GE.0) Then
   18       Continue
            quest='    NROW '
            ival=iiquest(quest,nr,'(i5)',0)
            If (nr.LE.0)Go To 18
            If (ival.EQ.-1)itest=-1
            If (nc*nr.GT.250000) Then
               nc=0
               nr=0
            Print *,' Error, grid dimensions too large nc*nr=<250000'
            Else
               itest=1
            End If
         Else
            itest=-1
         End If
   20 End Do
      If (itest.EQ.-1)Go To 100
C
C - Enter lower left corner x-coordinate of grid (xo).
C
      quest='    xo'
      ival=irquest(quest,xo,'(e16.8)',0)
      If (ival.EQ.-1)Go To 100
C
C - Enter interval spacing in x direction (dx).
C
      quest='    dx'
      ival=irquest(quest,dx,'(e16.8)',0)
      If (ival.EQ.-1)Go To 100
C
C - Enter lower left corner y-coordinate of grid (yo).
C
      quest='    yo'
      ival=irquest(quest,yo,'(e16.8)',0)
      If (ival.EQ.-1)Go To 100
C
C - Enter interval spacing in y direction (dy).
C
      quest='    dy'
      ival=irquest(quest,dy,'(e16.8)',0)
      If (ival.EQ.-1)Go To 100
```

```
C
C - Ask for dval to initialize grid to
C
C     call askdvl(dval,itest)
C     if (itest.eq.-1) go to 100
C
C - Initialize grid to user's dval.
C
C     call intgrd(dval)
C
  100 Continue
      If (ival.EQ.-1)itest=-1
      Return
      End
C+
C ********************************************************
C   Curoff - turns graphics crosr-hair cursor off
C            'G0' - command to turn cursor off.
C-
C **************************************************************
      Subroutine curoff
C
      Call esccom('G0')
C
      Return
      End
C+
C **************************************************************
C  Curon - turns graphics cross-hair cursor on.
C          'G1' - command to turn cursor on.
C-
C **************************************************************
      Subroutine curon
C
      Call esccom('G1')
C
      Return
      End
C+
C ***************************************************************
C  defclr - Allows the changing of a color in the Envision color
C           table by specifying the red, green and blue components.
C
C           color  = ASCII character representing position of
C                    color in color table.  See COLOR.INF or
C                    DEFCLR.INF or FNDCLR.INF.
C                  = '0'-'9', and ':'-'?'.
C
C           ired   = Integer value of color gun intensity,
C           igreen   range is from 0-15, 0=off, 15=full on.
C           iblue
C
C-
C ***************************************************************
      Subroutine defclr(color,ired,igreen,iblue)
      Character color*1,cred*1,cgreen*1,cblue*1,com*5
C
      Call fndclr(cred,ired)
      Call fndclr(cgreen,igreen)
      Call fndclr(cblue,iblue)
C
      com='Q'//color//cred//cgreen//cblue
      Call esccom(com)
C
```

```
      Return
      End
C+
C ***************************************************************
C  delloc - Deletes all location information associated with
C          polygon npoly.
C-
C ***************************************************************
      Subroutine delloc(npoly,itest)
      Common /parameter/parm(100,10)
      Common /scroenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /inout/xin(100,100),yin(100,100),xout(100,100),
     &yout(100,100)
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
C
      itest=1
C
C - Test the input variables...
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Then
C
C - Zero out all!!! of the coordintate locations, parms and all
C   internal arrays...
C
         Do 10 i=1,nptmax
         xscr(npoly,i)=0.0
         yscr(npoly,i)=0.0
         xin(npoly,i)=0.0
         xout(npoly,i)=0.0
         yin(npoly,i)=0.0
         yout(npoly,i)=0.0
   10    Continue
C
         numply(npoly)=0
C
         xminbx(npoly)=0.0
         xmaxbx(npoly)=0.0
         yminbx(npoly)=0.0
         ymaxbx(npoly)=0.0
C
         Do 20 j=1,10
         parm(npoly,j)=0.0
   20    Continue
C
         jnkloc(ngbloc)=npoly
      Else
         Call wrtmsg(' Error in delloc, npoly out of range')
         itest=-1
      End If
      Return
      End
C+
C ***************************************************************
C  delply - Deletes a polygon from the model.
C-
C ***************************************************************
      Subroutine delply(itest)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /flags/mcflag,votflg
      Common /calc/ncont,cmin,cdel
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
C
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1,color
     &*1
      Character votflg*2,ans*2,mcflag*2,mcur*2
C
      If (ntotal.GT.0) Then
C
C - Print enhanced message.
C
         Call enhmsg('*** Delete polygon mode ***')
         itest=1
         npoly=0
C
C - Print help message if user wants verbose answers.
C
         If (votflg.EQ.'V')Call hlpdel
C
C - Let user pick polygon; return npoly,ncorn,x,y.
C
   10    Continue
         Call intply
         Call pckply(npoly,ncorn,x,y,ans,mcflag,ierr)
         If (ans.EQ.'N')Go To 10
         If (ans.EQ.'Q')Go To 100
C
C - Now that we are certain about which polygon will be
C   deleted, decrement the polygon counters and increment
C   the garbage collection pointers (ngbtop,ngbloc).
C
         ntotal=ntotal-1
         If (ntotal.LT.0)ntotal=0
         ngbtop=ngbtop+1
         ngbloc=ngbloc+1
C
C - Physically delete (set to zero) all reference pointers from
C   the topology array (deltpl) and all location coordinate
C   arrays (delloc).
C
         itest=1
         Call deltpl(npoly,itest)
         If (itest.EQ.1) Then
            Call setclr(black)
            Call newold
         Else
            ntotal=ntotal+1
            ngbloc=ngbloc-1
            ngbtop=ngbtop-1
            Call setclr(plyclr)
            Call oldnew
         End If
C
C - Now either undraw the polygon (setclr(black)) if topology
C   deletion was succeeded, or redraw the polygon (setclr(plyclr))
C   if deletion failed.
C
         Call setfil(open)
         Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
         Call setclr(plyclr)
         Call setfil(solid)
C
C - Delete the location info for npoly if topology deletion succeeded.
C
         If (npoly.GT.0.AND.itest.GE.1) Then
            Call delloc(npoly,itest)
            If (ntotal.EQ.0)Call wrtmsg('*** No more polygons ***')
         End If
  100    Continue
      Else
         ntotal=0
         Call wrtmsg(' Polygon list emtpy, (i.e., No polygons)')
      End If
C
      Return
      End
C+
C ***************************************************************
C  delpnt - Allows the deleting of individual point of a polygon.
C-
C ***************************************************************
      Subroutine delpnt(itest)
      Dimension xply(100),yply(100)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /parameter /purm(100,10)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /flags/mcflag,votflg
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nnrminz(5),
     &nrmaxz(5)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /commands/nmax,epsiln,delin,delout
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character votflg*2,mcflag*2,mcur*2,ans*2,ans2*2
C
      Call enhmsg('*** Delete point mode ***')
C
      itest=0
      If (ntotal.GT.0) Then
C
C - If cursor type has not been selected prompt for type.
C
         If (mcflag.EQ.'N') Then
            Call askmoc(mcur)
            If (mcur.EQ.'Q') Then
               itest=-1
            Else
               mcflag=mcur
            End If
         Else
            mcur=mcflag
         End If
         If (votflg.EQ.'V')Call hlpdpt
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
         Do 10 while(itest.EQ.0)
         iflag=0
      iflag2=0
      If (mcur.EQ.'N'.OR.mcur.EQ.'C') Then
C
C - Initialize the temp arrays.
C
         Call inttmp
C
C - Let user pick polygon, return npoly,ncorn,x,y.
C
         Call pckply(npoly,ncorn,x,y,ans,mcur,ierr)
C
C - Ask if corner of polygon picked should be used.
C
         If (ans.EQ.'Y') Then
            nbrpts=numply(npoly)
            If (nbrpts.EQ.3) Then
               Call wrtmsg(' Sorry, can''t delete a point, polygon
     & only has three corners.')
               ans2='Q'
               itest=-1
            Else
               Call askpnt(ans2)
               If (ans2.EQ.'N') Then
                  Call msgsp
                  Call retpnt(x,y,mcur,ierr2)
                  If (izoom.EQ.1)Call invers(x,y,x,y)
               End If
            End If
C
            If (ierr2.LE.-1.OR.ans2.EQ.'Q') Then
               itest=-1
               Go To 20
            End If
C
            Do 35 i=1,nbrpts
            xply(i)=xscr(npoly,i)
            yply(i)=yscr(npoly,i)
 35         Continue
            icon=1
            Call clspnt(ncorn,dist,x,y,xply,yply,nbrpts,delout,icon)
            If (ncorn.EQ.0.OR.ierr2.LT.0) Then
               itest=-1
               Go To 20
            End If
C
C - Store new polygon
C
            npnts=nbrpts-1
            ntemp=npnts
            If (ncorn.EQ.1) Then
               Do 30 i=1,npnts
               xtemp(i)=xscr(npoly,i+1)
               ytemp(i)=yscr(npoly,i+1)
 30            Continue
            Else If (ncorn.EQ.nbrpts) Then
               Do 40 j=1,npnts
               xtemp(j)=xscr(npoly,j)
               ytemp(j)=yscr(npoly,j)
 40            Continue
            Else
               Do 50 k=1,ncorn-1
               xtemp(k)=xscr(npoly,k)
               ytemp(k)=yscr(npoly,k)
 50            Continue
```

```
                  Do 50 i=ncorn+1,nbrpts
                  xtemp(i-1)=xscr(npoly,i)
                  ytemp(i-1)=yscr(npoly,i)
   50             Continue
               End If
C
C - Now undraw the connecting lines to ncorn.
C
               If (ncorn.EQ.1) Then
                  x1=xscr(npoly,nbrpts)
                  y1=yscr(npoly,nbrpts)
                  x3=xscr(npoly,2)
                  y3=yscr(npoly,2)
               Else If (ncorn.EQ.nbrpts) Then
                  x1=xscr(npoly,nbrpts-1)
                  y1=yscr(npoly,nbrpts-1)
                  x3=xscr(npoly,1)
                  y3=yscr(npoly,1)
               Else
                  x1=xscr(npoly,ncorn-1)
                  y1=yscr(npoly,ncorn-1)
                  x3=xscr(npoly,ncorn+1)
                  y3=yscr(npoly,ncorn+1)
               End If
               x2=xscr(npoly,ncorn)
               y2=yscr(npoly,ncorn)
C
               If (izoom.EQ.1) Then
                  Call trans(x1,y1,x1,y1)
                  Call trans(x2,y2,x2,y2)
                  Call trans(x3,y3,x3,y3)
               End If
C
               Call setclr(black)
               Call drwlin(x1,y1,x2,y2)
               Call drwlin(x2,y2,x3,y3)
               Call setclr(white)
               Call drwlin(x1,y1,x3,y3)
C
C - Test the new polygon in the xtemp,ytemp array to see if it
C   is self-crossing.
C
               Call selftest(iflag3,nside1,nside2,xtemp,ytemp,ntemp)
               If (iflag3.EQ.-1) Then
                  Call wrtmsg(' Error, polygon is self-crossing')
                  iflag2=1
                  Go To 20
               End If
C
C - Test the new polygon in the xtemp,ytemp array and fit into
C   the topology structure.
C
               Call testopo(npoly,itest2)
               If (itest2.GE.0) Then
                  iflag=1
               Else
                  iflag2=1
               End If
               itest=1
C
   20          Continue
               If (iflag.EQ.0) Then
                  Call setclr(plyclr)
                  Call setfil(open)
                  Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
                  If (iflag2.eq.1) Then
                     Call setclr(black)
                     Call drwlin(x1,y1,x3,y3)
                     Call setclr(plyclr)
                     Call setfil(solid)
                  End If
               End If
            Else If (ans.EQ.'N') Then
               itest=0
            Else
               itest=-1
            End If
            Call setfil(solid)
            If (ierr.EQ.-1)itest=-1
         Else If (mcur.EQ.'Q') Then
            itest=-1
         End If
   10    End Do
      Else
         Call wrtmsg('  Sorry, no polygons')
         itest=-1
      End If
C
      Return
      End
C+
C **************************************************************
C  deltpl - Deletes a node (polygon npoly) from the topology
C           structure in the newtopo common blocks, and makes
C           connecting links for the remaining nodes.
C-
C **************************************************************
      Subroutine deltpl(npoly,itest)
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /commands/nmax,epsiln,delin,delout
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
C
C - Test to make sure that npoly is within range.
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Then
         iflag=0
         If (itest.EQ.2)iflag=1
C
C - Find the position (ndpstn) of the polygon (npoly) in the
C   tree (topology).  Abort and give user message if not
C   found, this means something is screwed up in the array
C   infnew (or entire topology array).
C
         Call fndply(ndpstn,npoly)
         If (ndpstn.LE.0) Then
            Print *,' Ndpstn =',ndpstn
            itest=-1
            Call wrtmsg(' You have big problems')
            Go To 100
         End If
         itest=1
C
C - Initialize the neighbor pointers of polygon npoly.
C
         iup=iupnew(ndpstn)
         idwn=idwnew(ndpstn)
         lft=ilfnew(ndpstn)
      irgt=irtnew(nCpsin)
C
C - Determine if the given polygon npoly is a left node of the
C   tree (lft=0).  If lft=0, then test to see if there is a
C   replacement polygon, first look down (idwn), if this is zero,
C   then look to the right (irgt).  If irgt is also zero this
C   means that there is no replacement polygon (irep)
C
      If (lft.EQ.0) Then
         irep=0
         If (idwn.GT.0)irep=idwn
C
         If (irep.GT.0) Then
            If (iup.GT.0)idwnew(iup)=irep
            ilfnew(irep)=0
            itemp=irep
  20        Continue
            iupnew(itemp)=iup
            ilast=itemp
            itemp=irtnew(itemp)
            If (itemp.GT.0)Go To 20
            irtnew(ilast)=irgt
            If (irgt.GT.0)ilfnew(irgt)=ilast
            itest=1
C
         Else If (irep.EQ.0) Then
            If (irgt.GT.0) Then
               If (iup.GT.0) Then
                  idwnew(iup)=irgt
                  ilfnew(irgt)=0
                  itest=1
               Else If (iup.EQ.0) Then
                  ilfnew(irgt)=0
                  itest=1
               Else
                  itest=-1
                  Call wrtmsg(' Problem with iup')
               End if
            Else If (irgt.EQ.0) Then
               If (iup.GT.0) Then
                  idwnew(iup)=0
                  itest=1
               Else If (iup.EQ.0) Then
                  itest=1
                  If (iflag.EQ.0)Call wrtmsg(
   &              ' *** No more Polygons ***')
               Else
                  itest=-1
                  Call wrtmsg(' Problem with iup')
               End If
            Else
               itest=-1
               Call wrtmsg(' Problem with irgt')
            End If
         Else
            itest=-1
            Call wrtmsg(' Problem with irep')
         End If
C
C - If the given polygon npoly is not a left node (i.e. lft.ne.0)
C   test for a replacement polygon (irep) by first looking down.
C
      Else If (lft.GT.0) Then
         irep=0
```

```
            If (idwn.GT.0)irep=idwn
            If (irep.GT.0) Then
               ilfnew(irep)=lft
               irtnew(lft)=irep
               itemp=irep
   30          Continue
               iupnew(itemp)=iup
               ilast=itemp
               itemp=irtnew(itemp)
               If (itemp.GT.0)Go To 30
               irtnew(ilast)=irgt
               If (irgt.GT.0)ilfnew(irgt)=ilast
               itest=1
            Else If (irep.EQ.0) Then
               irtnew(lft)=irgt
               If (irgt.GT.0)ilfnew(irgt)=lft
               itest=1
            Else
               itest=-1
               Call wrtmsg(' Problem with irep')
            End If
         Else
            itest=-1
            Call wrtmsg(' Problem with lft')
         End If
C
C - Zero out the deleted polygon's structure from the tree and
C   add the deleted position (ndpstn) to the garbage table at
C   position igarb.
C
         If (itest.GE.0) Then
            infnew(ndpstn)=0
            iupnew(ndpstn)=0
            idwnew(ndpstn)=0
            ilfnew(ndpstn)=0
            irtnew(ndpstn)=0
C
            jnktop(ngbtop)=ndpstn
         End IF
C
  100    Continue
      Else
         Call wrtmsg(' Error in DELTPL, npoly not in range
     & 1=<npoly=<nmax')
         itest=-1
      End If
C
      Return
      End
C+
C ***************************************************************
C  displa - Selects the drawing, erase, and redraw modes
C           (default=122).  For Envision model 230 only.
C-
C ***************************************************************
      Subroutine displa
C
      Call esccom('OG122')
C
      Return
      End
C+
C ***************************************************************
C  drawwalk - Walks the topology structure (tree) and clips and
```

```
C              draws the polygon encountered.
C-
C *************************************************************
      Subroutine drawwalk
      Dimension xpoly(100),ypoly(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /topology/info(100),iupper(100),idown(100),ileft(100),
     &iright(100)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
C
      Call fndtop(ntop)
      If (ntop.GE.1) Then
C
         next=ntop
         ngon=ntop
         itest=0
C
         Do 10 while(itest.EQ.0)
         npoly=info(next)
         If (npoly.GE.1) Then
            nbrpts=numply(npoly)
            If (nbrpts.GT.0) Then
               If (izoom.EQ.1) Then
                  Do 20 i=1,nbrpts
                  Call trans(xpoly(i),ypoly(i),xscr(npoly,i),
     &            yscr(npoly,i))
   20             Continue
               Else
                  Do 25 j=1,nbrpts
                  xpoly(j)=xscr(npoly,j)
                  ypoly(j)=yscr(npoly,j)
   25             Continue
               End If
               Call drwclp(xpoly,ypoly,nbrpts,xleft,xright,ybot,ytop)
               If (itest2.GE.0) Then
                  Call walk(next,ngon,itest3)
                  If (next.GE.1) Then
                     ngon=next
                  Else If (next.EQ.0.OR.itest3.EQ.0) Then
                     itest=1
                  Else If (next.LT.0.OR.itest3.EQ.-1) Then
                     Print *,' Error in walking tree at position',ngon
                     itest=-1
                  End If
               End If
            Else
               itest=-1
            End If
         Else
            Print *,' Error, npoly out of range'
            itest=-1
         End If
   10    End Do
      Else
      Print *,' Error ntop not found or out of range'
      End If
C
      Return
      End
C+
C ***********************.**************************************
C  drwbox - DRaW BoX - Draw a box with the boundaries determined
```

```
C           by the world coordinate pair (x0,y0) and (x1,y1).
C-
C **************************************************************
      Subroutine drwbox(x0,y0,x1,y1)
      Character com*12,wcbp*5
C
      ix0=jnint(x0)
      iy0=jnint(y0)
      ix1=jnint(x1)
      iy1=jnint(y1)
      com='UX'//wcbp(ix0,iy0)//wcbp(ix1,iy1)
      Call esccom(com)
C
      Return
      End
C+
C **************************************************************
C  drwclp - DRaW CLipped Polygon - Clips and draws the polygon
C           xpoly,ypoly on the Envision terminal.
C-
C **************************************************************
      Subroutine drwclp(xpoly,ypoly,nbrpts,xleft,xright,ybot,ytop)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
C
      Do 10 i=1,nbrpts
      If (i.EQ.nbrpts) Then
         x1=xpoly(i)
         y1=ypoly(i)
         x2=xpoly(1)
         y2=ypoly(1)
      Else
         x1=xpoly(i)
         y1=ypoly(i)
         x2=xpoly(i+1)
         y2=ypoly(i+1)
      End If
C
      Call tstend(inout,x1,y1,x2,y2,xleft,xright,ybot,ytop)
C
      If (inout.EQ.3) Then
         Call drwlin(x1,y1,x2,y2)
      Else If (inout.EQ.2) Then
         Call clipper(xone,yone,x1,y1,x2,y2,xleft,xright,
     &   ybot,ytop)
         Call drwlin(xone,yone,x2,y2)
      Else If (inout.EQ.1) Then
         Call clipper(xtwo,ytwo,x2,y2,x1,y1,xleft,xright,
     &   ybot,ytop)
         Call drwlin(x1,y1,xtwo,ytwo)
      Else If (inout.EQ.0) Then
         Call clipper(xone,yone,x1,y1,x2,y2,xleft,xright,
     &   ybot,ytop)
         Call clipper(xtwo,ytwo,x2,y2,x1,y1,xleft,xright,
     &   ybot,ytop)
         Call drwlin(xone,yone,xtwo,ytwo)
      End If
   10 Continue
C
      Return
      End
C+
C **************************************************************
C  drwlin - Draw a line connecting the world coordinates (x0,y0)
C           and (x1,y1).
```

```
C-
C **************************************************************
      Subroutine drwlin(x0,y0,x1,y1)
      Charactor com*12,wcbp*5
C
      ix0=jnint(x0)
      iy0=jnint(y0)
      ix1=jnint(x1)
      iy1=jnint(y1)
C
      com='OV'//wcbp(ix0,iy0)//wcbp(ix1,iy1)
      Call esccom(com)
C
      Return
      End
C+
C **************************************************************
C  drwpln - DRaW Poly LiNe - Draws a polyline figure connecting
C           world coordinates (xpts,ypts) in the order passed to
C           drwpln.  See the Envision reference manual or drw-
C           pln.inf for details.
C-
C **************************************************************
      Subroutine drwpln(xpts,ypts,nbrpts)
      Dimension xpts(nbrpts),ypts(nbrpts)
      Character esc*1,ncbp*3,wcbp*5
C
      esc=char(27)
      Print *,esc,'Om',ncbp(nbrpts),(wcbp(jnint(xpts(i)),
     &jnint(ypts(i))),i=1,nbrpts)
C
      Return
      End
C+
C **************************************************************
C  drwply - Draws a polygon on the Envision terminal.  Connects
C
C           xpts,ypts - Array containing the world coordinates
C                       of polygon vertices.
C
C           nbrpts - Number of vertices in xpts,ypts.
C-
C **************************************************************
      Subroutine drwply(xpts,ypts,nbrpts)
      Dimension xpts(nbrpts),ypts(nbrpts)
      Character esc*1,ncbp*3,wcbp*5
C
      esc=char(27)
      Print *,esc,'OM',ncbp(nbrpts),(wcbp(jnint(xpts(i)),
     &jnint(ypts(i))),i=1,nbrpts)
C
      Return
      End
C+
C **************************************************************
C - drwpnt - DRaW PoiNT - Draw a point at world coordinates
C           (x0,y0).
C-
C **************************************************************
      Subroutine drwpnt(x0,y0)
      Character com*7,wcbp*5
C
      ix0=jnint(x0)
      iy0=jnint(y0)
      com='00'//wcbp(ix0,iy0)
      Call esccom(com)
C
      Return
      End
C+
C **************************************************************
C  edtply - EDIT PoLY - Driver for Edit polygon mode.
C          Options are:
C
C               a = Add points
C               d = Delete a point
C               m = Move a point
C               h = Help
C               q = Quit and return to Polygon mode
C
C-
C **************************************************************
      Subroutine edtply(itest)
      Character quest*80,ans*2
C
      Call enhmsg('*** Edit polygon mode ***')
C
      itest=0
      ans='h'
      Do 10 while(itest.EQ.0)
         quest=' Edit polygon mode (a/d/m/h/q)'
         ival=iaquest(quest,ans,'(a2)',2)
C
         If (ans.EQ.'A'.OR.ans.EQ.'a') Then
            Call addpnt(ierror)
         Else If (ans.EQ.'M'.OR.ans.EQ.'m') Then
            Call movpnt(ierror)
         Else If (ans.EQ.'D'.OR.ans.EQ.'d') Then
            Call delpnt(ierror)
         Else If (ans.EQ.'H'.OR.ans.EQ.'h') Then
            Call hlpedt
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1.OR.ans.EQ.'q') Then
            itest=1
         Else
            Call errmsg
         End If
C        if (ierror.eq.-1) itest=-1
         ans='q'
   10 End Do
C
      Return
      End
C+
C **************************************************************
C  enhmsg - Displays enhanced message at user's terminal.
C-
C **************************************************************
      Subroutine enhmsg(text)
      Character text*(*),black*1,white*1
C
      white='1'
      black='0'
      Call envbcl(white)
      Call wrtmsg(text)
      Call envbcl(black)
C
      Return
      End
```

```
C+
C ***************************************************************
C  entmou - Enables the mouse, draws polygon as user enters it,
C           and returns polygon coordinates via xscrn,yscrn.
C-
C ***************************************************************
      Subroutine entmou(xscrn,yscrn,nbrpts,itest)
      Dimension xscrn(100),yscrn(100)
      Character ans*2
C
      Call curon
      Call softky('1')
      Call setmou
      Call loadmou
      nbrpts=0
      itest=0
      i=0
      If (itest.EQ.0) Then
   10    Continue
         Call getmou(mode,ix,iy)
         x=float(ix)
         y=float(iy)
         If (mode.EQ.1.OR.mode.EQ.2) Then
            If (i.EQ.99.AND.mode.EQ.1) Then
               Print *,
     &         ' Last point, only button 2 or buttons 2&3 allowed'
            Else
               i=i+1
               xscrn(i)=x
               yscrn(i)=y
               Call drwpnt(x,y)
               If ((i.GT.1).AND.(i.LE.100)) Then
                  If ((i.EQ.2).AND.(mode.EQ.2)) Then
                     Call asktwz(ans)
                     If (ans.EQ.'Q') Then
                        nbrpts=i
                        itest=-1
                     Else
                        mode=1
                     End If
                  End If
                  If (itest.EQ.0) Then
                     Call drwlin(xscrn(i-1),yscrn(i-1),xscrn(i),yscrn(i
     &               ))
                     If (mode.EQ.2) Then
                        Call drwlin(xscrn(i),yscrn(i),xscrn(1),yscrn(1)
     &                  )
                        nbrpts=i
                        itest=1
                     End If
                  End If
               End If
            End If
C
         Else If (mode.EQ.3) Then
            Call hlpmou
C
         Else If (mode.EQ.23) Then
            nbrpts=i
            itest=-1
C
         Else
            Call errmsg
C
```

```
         End If
         If (i.EQ.99.AND.itest.EQ.0) Then
            Print *,' 99 coordinates entered, only one more allowed'
         End If
         If (i.GT.100)itest=-1
         If (itest.EQ.0)Go To 10
      End If
C
      Call softky('0')
      Call curoff
      Return
      End
C+
C ***************************************************************
C  entply - Enables cursor keys, draws polygon as user enters it,
C           and returns coordinates via xscrn,yscrn.
C-
C ***************************************************************
      Subroutine entply(xscrn,yscrn,nbrpts,itest)
      Dimension xscrn(100),yscrn(100)
      Character ans*1,ans1*1
C
      Call curon
      nbrpts=0
      itest=0
      i=0
      If (itest.EQ.0) Then
   10    Continue
         Call gtpnt(ans,x,y)
         If (ans.EQ.'e'.OR.ans.EQ.'f') Then
            If (i.EQ.99.AND.ans.EQ.'e') Then
               Print *,' Last point, only (f/q) allowed'
            Else
               i=i+1
               xscrn(i)=x
               yscrn(i)=y
               Call drwpnt(x,y)
               If ((i.GT.1).AND.(i.LE.100)) Then
                  If ((i.EQ.2).AND.(ans.EQ.'f')) Then
                     Call asktwz(ans1)
                     ans=ans1
                     If (ans.EQ.'q') Then
                        nbrpts=i
                        itest=-1
                     End If
                  End If
                  If (itest.EQ.0) Then
                     Call drwlin(xscrn(i-1),yscrn(i-1),xscrn(i),yscrn(i
     &               ))
                     If (ans.EQ.'f') Then
                        Call drwlin(xscrn(i),yscrn(i),xscrn(1),yscrn(1)
     &                  )
                        nbrpts=i
                        itest=1
                     End If
                  End If
               End If
            End If
C
         Else If (ans.EQ.'h') Then
            Call hlpent
C
         Else If (ans.EQ.'q') Then
            nbrpts=i
            itest=-1
C
         Else
            Call errmsg
C
         End If
         If (i.EQ.99.AND.itest.EQ.0) Then
            Print *,' 99 coordinates entered, only one more allowed'
         End If
         If (i.GT.100)itest=-1
         if (itest.EQ.0)Go To 10
      End If
C
      Call curoff
      Return
      End
C+
C ***************************************************************
C  envbcl - Sets the alphanumeric display background color.
C
C           See ENVBCL.INF and/or the Envision reference manual.
C-
C ***************************************************************
      Subroutine envbcl(color)
      Character com*2,color*1
C
      com='b'//color
      Call esccom(com)
C
      Return
      End
C+
C ***************************************************************
C  envclr - Automatically loads in the Envsion terminal and picks
C           up to 13 colors in a spectral color sequence.  1=white,
C           2=green, 3-ncolors=spectrum, rest=black.  Color spectrum
C           is left justified at iclr=3 if nshit=0.
C
C           Author: Robert Simpson
C-
C ***************************************************************
      Subroutine envclr(iounit,ncolors,nshift)
      Character color*1,v2c*1
      Integer ir(0:15),ig(0:15),ib(0:15)
      Parameter (nmax=14)
      Integer ird(0:15),igr(0:15),ibl(0:15)
      Data ird/00,09,07,00,00,00,00,11,15,15,15,15,15,15,15,15/
      Data igr/00,00,00,00,10,13,14,15,15,12,09,07,00,00,00,15/
      Data ibl/00,09,11,15,14,10,06,00,00,00,00,00,00,12,15,15/
C
      ncolors=max(2,ncolors)
      ncolors=min(13,ncolors)
      nshift=max(0,nshift)
      nshift=min(13-ncolors,nshift)
C
C     Set all black...
C
      Do 20 i=0,15
      ir(i)=0
      ig(i)=0
      ib(i)=0
   20 Continue
C
C     Set 1= white
```

```
C
      ir(1)=15
      ig(1)=15
      ib(1)=15
C
C     Set 2= cursor green...
C
      ir(2)=0
      ig(2)=15
      ib(2)=0
C
C     Fill ncolors into 3 thru ncolors+2 by interpolating in spectrum of 14...
C
      Do 40 i=1,ncolors
      clr=Real (i-1)*Real (nmax-1)/Real (ncolors-1)+1.0
      Do 40 j=1,nmax-1
      If (clr.GE.Real (j).AND.clr.LE.Real (j+1)) Then
         ir(i+2+nshift)=jnint((ird(j+1)-ird(j))*(clr-Real (j))+ird(j))
         ig(i+2+nshift)=jnint((igr(j+1)-igr(j))*(clr-Real (j))+igr(j))
         ib(i+2+nshift)=jnint((ibl(j+1)-ibl(j))*(clr-Real (j))+ibl(j))
      End If
   40 Continue
C
C     Enhance yellow...
C
      Do 50 i=3,ncolors+2
      If (ib(i).EQ.0.AND.ig(i).GE.14.AND.ir(i).GE.12) Then
         ir(i)=15
         ig(i)=15
      End If
   50 Continue
C
C     Write color spectrum...
C
      Do 70 iclr=0,15
      Call fndclr(color,iclr)
      Call defclr(color,ir(iclr),ig(iclr),ib(iclr))
   70 Continue
C
      Return
      End
C+
C ****************************************************************
C  errmod - Prints error message at user's terminal if output file
C           is requested before other conditions are satisfied.
C-
C ****************************************************************
      Subroutine errmod
C
      Write (6,10)
   10 Format (/,' Error output files can not be generated until',
     &' model file is created',/)
C
      Return
      End
C+
C ****************************************************************
C  errmsg - Prints message at user's terminal if a wrong answer
C           is entered.
C-
C ****************************************************************
      Subroutine errmsg
C
      Write (6,10)
   10 Format (/,' Wrong answer...try again')
C
      Return
      End
C+
C ***************************************************************
C  esccom - Sends the character string 'com' to the Envision terminal.
C-
C ***************************************************************
      Subroutine esccom(com)
      Character*(*) com,esc*1
      Parameter (esc=char(27))
C
      ileng=len(com)
      Write (6,20)esc//com
   20 Format (x,a(ileng+1))
C
      Return
      End
C+
C ***************************************************************
C  fndbbx - Finds the bounding box around the polygon passed in
C           the arrays xpoly,ypoly.  Adds a boundary (delta)
C           to the boundary xmin,xmax,ymin,ymax returned.
C-
C ***************************************************************
      Subroutine fndbbx(xmin,xmax,ymin,ymax,xpoly,ypoly,nbrpts,delta)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
C
      delta=abs(delta)
C
C - Initialize the maximum and minimum values of xmin,xmax,
C   ymin,ymax.
C
      xmin=xpoly(1)
      xmax=xpoly(1)
      ymin=ypoly(1)
      ymax=ypoly(1)
C
C - Go through the set of points defining the polygon and
C   find the smallest (xmin,ymin) and largest (ymin,ymax).
C
      Do 10 i=1,nbrpts
      xmin=amin1(xpoly(i),xmin)
      xmax=amax1(xpoly(i),xmax)
      ymin=amin1(ypoly(i),ymin)
      ymax=amax1(ypoly(i),ymax)
   10 Continue
C
C - Test the boundary limits for over/under flow when the
C   epsilon (delta) is added/subtracted.
C
      xmin=xmin-delta
      xmax=xmax+delta
      ymin=ymin-delta
      ymax=ymax+delta
C
      Return
      End
C+
C ***************************************************************
C  fndbpl - Finds an inner and outer bounding polygon around poly-
C           gon npoly.
C-
```

```
C ****************************************************************
      Subroutine fndbpl(npoly,itest)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /inout/xin(100,100),yin(100,100),xout(100,100),
     & yout(100,100)
      Common /commands/nmax,epsiln,delin,delout
      Dimension xintmp(100),yintmp(100),xoutmp(100),youtmp(100),
     & xpoly(100),ypoly(100)
C
      itest=0
      nbrpts=numply(npoly)
      If (nbrpts.LE.2) Then
         itest=-1
      Else
         Do 5 ii=1,nbrpts
         xpoly(ii)=xscr(npoly,ii)
         ypoly(ii)=yscr(npoly,ii)
    5    Continue
C
         Call boundpoly(itest,xintmp,yintmp,xoutmp,youtmp,xpoly,
     & ypoly,nbrpts,delin,delout)
         If (itest.EQ.1) Then
            Do 10 i=1,nbrpts
            xin(npoly,i)=xintmp(i)
            yin(npoly,i)=yintmp(i)
            xout(npoly,i)=xoutmp(i)
            yout(npoly,i)=youtmp(i)
 10         Continue
         End If
      End If
C
      Return
      End
C+
C ***************************************************************
C  fndcde - FIND CoDE - Returns the code for the region, ex-
C           pressed as ibit4,ibit3,ibit2,ibit1, of the screen
C           location (x,y).  See Figure 5-5, p.66, "Principles
C           of Interactive Computer Graphics", by Newman and
C           Sproull, 1979.
C
C           ibit1 = 1, if x < xleft
C           ibit2 = 1, if x > xright
C           ibit3 = 1, if y < ybot
C           ibit4 = 1, if y > ytop
C
C-
C ********************* ****************************************
      Subroutine fndcde(ibit4,ibit3,ibit2,ibit1,x,y,xleft,xright,
     &ybot,ytop)
C
      ibit4=0
      ibit3=0
      ibit2=0
      ibit1=0
C
      If (x.LT.xleft) Then
         ibit1=1
      Else If (x.GT.xright) Then
         ibit2=1
      End If
C
      If (y.LT.ybot) Then
         ibit3=1
      Else If (y.GT.ytop) Then
         ibit4=1
      End If
C
      Return
      End
C+
C *************************************************************
C  fndclr - Converts a numeric code for color in the color table
C           into the equivalent ASCII (hexadecimal) code for the
C           Envision terminal.  See also COLOR.INF, or FNDCLR.INF.
C
C           ncolor - Position number of color in color table 0-15.
C                  = 0 - usually black
C                  = 1-15 - Same as color mapping 1-'?', see
C                    Envision reference manual.
C           color - Ascii code (a1) for color in color table
C                  = '0'-'9',':'-'?'.
C                  = '0', if ncolor<0 or ncolor>15, an error
C                    message is printed.
C-
C *************************************************************
      Subroutine fndclr(color,ncolor)
      Character color*1
C
      If (ncolor.GE.0.AND.ncolor.LE.15) Then
         color=char(ncolor+ichar('0'))
      Else
         color='0'
         Call wrtmsg(' Error, value past to FNDCLR out of range.')
      End If
C
      Return
      End
C+
C ************************************************************
C  fndnod - FIND NODe - Finds the currently available node in
C           the tree.  Starts with jnktop arrays and then scans
C           through the topology arrays.  If all nodes have been
C           used ndpstn=0.
C
C           ndpstn = node position
C                  = 1 to nmax
C                  = 0 no more nodes
C                  =-1 error condition
C
C-
C *************************************************************
      Subroutine fndnod(ndpstn)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /commands/nmax,epsiln,delin,delout
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
C
      itest=0
    5 Continue
      ndpstn=0
      If (ntotal.GE.0.AND.ntotal.LE.nmax) Then
         If (ntotal.EQ.nmax) Then
            Print *,' Polygon list full, only',nmax,'polygons allowed'
         Else
            If (ngbtop.GT.0) Then
               ndpstn=jnktop(ngbtop)
               jnktop(ngbtop)=0
               ngbtop=ngbtop-1
            Else If (ngbtop.EQ.0) Then
               i=1
               Do 10 while(ndpstn.EQ.0)
                  If (info(i).LE.0) Then
                     ndpstn=i
                  Else
                     i=i+1
                  End If
                  If (i.GT.nmax)ndpstn=-1
   10          End Do
            Else
               itest=itest+1
               ngbtop=0
               Do 20 j=1,nmax
               If (jnktop(j).GT.0)ngbtop=ngbtop+1
   20          Continue
               If (itest.EQ.1) Then
                  Call wrtmsg(' Error in ngbtop, recovery attempted')
               Else If (itest.GE.2) Then
                  ndpstn=-1
                  Call wrtmsg(' Recovery attempt failed')
               End If
            End If
         End If
      Else
         ndpstn=-1
         Call wrtmsg(' Error in ntotal, out of range')
      End If
      If (itest.EQ.1)Go To 5
C
      Return
      End
C+
C ***************************************************************
C  fndnum - Finds the next available polygon position number.
C           Starts with the jnkloc arrays and then scans the
C           numply (NUMBER points in POLYGON) array for an open
C           position (i.e. numply(npoly)=0).
C
C           npoly = Number of polygon position
C                 = 1 to nmax
C                 = 0 no more polygons spaces
C                 =-1 error condition
C-
C ***************************************************************
      Subroutine fndnum(npoly)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /commands/nmax,epsiln,delin,delout
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
C
      itest=0
    5 Continue
      npoly=0
      If (ntotal.GE.0.AND.ntotal.LE.nmax) Then
         If (ntotal.EQ.nmax) Then
            Print *,' Polygon list full, only',nmax,'polygons allowed'
         Else
            If (ngbloc.GT.0) Then
               npoly=jnkloc(ngbloc)
               jnkloc(ngbloc)=0
               ngbloc=ngbloc-1
            Else If (ngbloc.EQ.0) Then
```

```
               i=1
               Do 10 while(npoly.EQ.0)
                  If (numply(i).LE.0) Then
                     npoly=i
                  Else
                     i=i+1
                  End If
                  If (i.GT.nmax)npoly=-1
   10          End Do
            Else
               itest=itest+1
               ngbloc=0
               Do 20 j=1,nmax
               If (jnkloc(j).GT.0)ngbloc=ngbloc+1
   20          Continue
               If (itest.EQ.1) Then
                  Call wrtmsg('  Error in ngbloc, recovery attempted')
               Else If (itest.GE.2) Then
                  npoly=-1
                  Call wrtmsg('  Recovery attempt failed')
               End If
            End If
         End If
      Else
         npoly=-1
         Call wrtmsg('  Error in ntotal')
      End If
      If (itest.EQ.1)Go To 5
C
      Return
      End
C+
C **************************************************************
C  fndply - Finds and returns the node position (ndpstn) of
C           polygon npoly in the tree.
C
C           ndpstn = Node position of polygn npoly
C                  > 0, Node for npoly exists
C                  =-1, Node for npoly does not exist
C
C-
C **************************************************************
      Subroutine fndply(ndpstn,npoly)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /commands/nmax,epsiln,delin,delout
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Than
         i=1
         ndpstn=0
         Do 20 while(ndpstn.EQ.0)
            If (info(i).EQ.npoly) Then
               ndpstn=i
            Else
               i=i+1
               If (i.GT.nmax)ndpstn=-1
            End If
   20    End Do
      Else
         Call wrtmsg('  Error, npoly passed to FNDPLY out of range.')
         ndpstn=-1
      End If
C
      Return
      End
C+
C ***************************************************************
C  fndscn - Converts the pair of grid locations xgrid,ygrid into
C           there corresponding screen locations xscrn,yscrn.
C-
C ***************************************************************
      Subroutine fndscn(xscrn,yscrn,xgrid,ygrid,nbrpts)
      Dimension xgrid(2),ygrid(2),xscrn(2),yscrn(2)
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
C
      Do 50 i=1,nbrpts
      xscrn(i)=xinit+(xgrid(i)-xstart)*xsc
      yscrn(i)=yinit+(ygrid(i)-ystart)*ysc
   50 Continue
C
      Return
      End
C+
C **************************************************************
C  fndsid - Finds the corners (ncorn,ncorn2) of the side of the
C           polygon npoly that is closest to the point xtest,ytest.
C-
C **************************************************************
      Subroutine fndsid(ncorn,ncorn2,dmin,xtest,ytest,npoly,itest)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /inout/xin(100,100),yin(100,100),xout(100,100),
     &yout(100,100)
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Parameter (vaxmin=-1.7e+38,vaxmax=1.7e+38)
      Dimension xpoly(6),ypoly(6)
C
      ncorn=0
      ncorn2=0
      itest=0
      iset=0
      dmin=vaxmax
      totald=vaxmax
C
C - Test if npoly is within range.
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Then
         nbrpts=numply(npoly)
         If (nbrpts.GE.1.AND.nbrpts.LE.nptmax) Then
            Do 10 i=1,nbrpts
            iup=i+1
            If (i.EQ.nbrpts)iup=1
C
            xpoly(1)=xscr(npoly,i)
            xpoly(2)=xout(npoly,i)
            xpoly(3)=xout(npoly,iup)
            xpoly(4)=xscr(npoly,iup)
            xpoly(5)=xin(npoly,iup)
            xpoly(6)=xin(npoly,i)
C
            ypoly(1)=yscr(npoly,i)
            ypoly(2)=yout(npoly,i)
            ypoly(3)=yout(npoly,iup)
            ypoly(4)=yscr(npoly,iup)
            ypoly(5)=yin(npoly,iup)
            ypoly(6)=yin(npoly,i)
C
C - Now test if the point xtest,ytest is in the polygon bounded
```

```
C   by xpoly,ypoly.
C
            Call plytst(xpoly,ypoly,6,xtest,ytest,inout)
C
            If (inout.EQ.1) Then
               totald=(xscr(npoly,i)-xtest)**2+(yscr(npoly,i)-
     &         ytest)**2+(xscr(npoly,iup)-xtest)**2+(yscr(npoly,iup)-
     &         ytest)**2
               If (totald.LT.dmin) Then
                  ncorn=i
                  ncorn2=iup
                  dmin=totald
                  iset=1
               End If
            End If
   10       Continue
C
            If (iset.EQ.1) Then
               itest=1
            Else
               Print *,' Error, could not find polygon''s side'
               itest=-1
            End If
         Else
            Print *,' Error, nbrpts out of range'
            itest=-1
         End If
      Else
         Print *,' Error, npoly out of range'
         itest=-1
      End If
C
      Return
      End
C+
C *************************************************************
C  fndtop - FIND TOP - Finds the top node or root (ntop) of the
C           tree in the topology arrays.
C-
C *************************************************************
      Subroutine fndtop(ntop)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /commands/nmax,epsiln,delin,delout
C
      ntop=0
      i=1
C
      Do 10 while(ntop.EQ.0)
         If (info(i).GE.1.AND.info(i).LE.nmax.AND.iupper(i).EQ.0
     &.AND.ileft(i).EQ.0) Then
            ntop=i
         Else
            i=i+1
            If (i.GT.nmax)ntop=-1
         End If
   10 End Do
C
      Return
      End
C+
C *************************************************************
C  fndtopnew - FIND TOP NEW- Finds the top node or root (ntop) of the
C           tree in the newtopo arrays.
```

```
C-
C ******************************************************************
      Subroutine fndtopnew(ntop)
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /commands/nmax,epsiln,delin,delout
C
      ntop=0
      i=1
C
      Do 10 while(ntop.EQ.0)
         If (infnew(i).GE.1.AND.infnew(i).LE.nmax.AND.iupnew(i).EQ.0
     &.AND.ilfnew(i).EQ.0) Then
           ntop=i
        Else
           i=i+1
           If (i.GT.nmax)ntop=-1
        End If
   10 End Do
C
      Return
      End
C+
C ******************************************************************
C  fndtpl - Finds the and sets the topology of the polygon npoly
C           the corner locations are passed to the test routines
C           in the temp common block.
C
C           The topology is an implementation of a quadruply-linked
C           list [Knuth, "Fundamental Algorithms", p. 352].
C
C-
C ******************************************************************
      Subroutine fndtpl(npoly,ntop,itest)
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),
     &yscr(100,100)
      Common /neighbors/icnt,in(100),jcnt,jout(100)
      Common /state/lflast,lftin,lftout,ndpstn,iup
      Common /commands/nmax,epsiln,delin,delout
C
C
C - Test to see if this is the first polygon
C
      If (ntotal.EQ.0) Then
        Call fndnod(ndpstn)
        If (ndpstn.GE.1) Then
           infnew(ndpstn)=npoly
           iupnew(ndpstn)=0
           idwnew(ndpstn)=0
           ilfnew(ndpstn)=0
           irtnew(ndpstn)=0
           ntop=ndpstn
           itest=1
        Else
           Call wrtmsg(' Failed on ntotal=0 test')
           itest=-1
        End If
C
C - Start at top of tree... ntop
C
      Else If (ntotal.GT.0) Then
        ngon=ntop
```

```
         nhit=0
         icnt=0
         jcnt=0
         iup=0
         nleft=0
         lflast=0
         lftin=0
         lftout=0
         Do 5 i=1,nmax
            in(i)=0
            jout(i)=0
    5    Continue
C
C - Find the next available node position in the topology arrays.
C
         Call fndnod(ndpstn)
C
C - Clear the arrays associated with this node.
C
         If (ndpstn.GE.1) Then
            infnew(ndpstn)=0
            iupnew(ndpstn)=0
            idwnew(ndpstn)=0
            ilfnew(ndpstn)=0
            irtnew(ndpstn)=0
            itest=0
         Else
            itest=-1
         End If
C
C  Test npoly to see if it is within ngon
C
         Do 10 while(itest.EQ.0)
            nloc=infnew(ngon)
            icon=0
            Call tstpgn(inout,nloc,icon,ierror)
            If (ierror.LE.-1) Go To 100
            If (inout.EQ.0) Then
               Call tstxcr(ncross,nloc)
               If (ncross.EQ.1)inout=-1
            End If
C
            If (inout.EQ.1) Then
               nhit=1
               If (icnt.EQ.0)lftin=ngon
               icnt=icnt+1
               in(icnt)=ngon
               nleft=ngon
               ngon=irtnew(ngon)
               If (ngon.EQ.0) Then
C                 print *,' Test 1'
                  Call settpl(npoly)
                  itest=1
               End If
C
            Else If (inout.EQ.0) Then
               If (nhit.EQ.0) Then
                  icon=1
                  nloc=infnew(ngon)
                  Call tstpgn(inout2,nloc,icon,ierror2)
                  If (ierror2.LE.-1) Go To 100
                  If (inout2.EQ.0) Then
                     lflast=ngon
                     nleft=ngon
                     ngon=irtnew(ngon)
                     If (ngon.EQ.0) Then
C                       print *,' Test 2'
                        irtnew(nleft)=ndpstn
                        infnew(ndpstn)=npoly
                        iupnew(ndpstn)=iup
                        idwnew(ndpstn)=0
                        ilfnew(ndpstn)=nleft
                        irtnew(ndpstn)=0
                        itest=1
                     End If
                  Else If (inout2.EQ.1) Then
                     iup=ngon
                     ngon=idwnew(ngon)
                     If (ngon.EQ.0) Then
C                       print *,' Test 3'
                        idwnew(iup)=ndpstn
                        infnew(ndpstn)=npoly
                        iupnew(ndpstn)=iup
                        idwnew(ndpstn)=0
                        ilfnew(ndpstn)=0
                        irtnew(ndpstn)=0
                        itest=1
                     End If
                  Else If (inout2.EQ.-1) Then
                     itest=-1
                     Call wrtmsg(' Error, polygon crosses another')
                  Else
                     itest=-1
                     Call wrtmsg(' Error with inout2 value (not equal
     & to 1,0,-1)')
                  End If
C
               Else If (nhit.EQ.1) Then
                  If (jcnt.EQ.0)lftout=ngon
                  jcnt=jcnt+1
                  jout(jcnt)=ngon
                  nleft=ngon
                  ngon=irtnew(ngon)
                  If (ngon.EQ.0) Then
C                    Print *,' Test 4'
C -- Procedure to reset arrays.....
C
                     Call settpl(npoly)
                     itest=1
                  Else If (ngon.GT.0) Then
                     itest=0
                  Else
                     Call wrtmsg(' Problem with ngon (less than zero)')
                     itest=-1
                  End If
               End If
            Else If (inout.EQ.-1) Then
               Call wrtmsg(' Error, polygon crosses another')
               itest=-1
            Else
               Call wrtmsg(' Error in tstpgn (inout was not 1,0,-1)')
               itest=-1
            End If
   10    End Do
      End If
C
100   Continue
      If (ierror.LE.-1.OR.ierror2.LE.-1) itest=-1
      Return
      End
C+
C **************************************************************
C  getgrd - Asks the user for the name of grid (grdnam), and
C           reads in the grid.
C-
C ***************************************************************
      Subroutine getgrd(itest)
      Common /names/grdnam,modnam,modgrd
      Character quest*80,modgrd*80,modnam*80,grdnam*80
C
   10 Continue
      quest=' Grid name'
      Call asknam(grdnam,quest,itest)
      If (itest.EQ.-1)Go To 100
C
      Call redgrd(grdnam,itest)
      If (itest.EQ.-1)Go To 10
C
  100 Continue
      Return
      End
C+
C ************************************************************************
C  getlin - Returns the world coordinates (xline,yline) of a line
C           entered on the Envision terminal.  The cross-hair cursor
C           keys or the mouse may be used.
C
C           mcflag = Flag for defaulting to cross-hair cursor type
C                    (mcur takes on the same values as mcflag)
C                  = C = Cross-hair cursor keys
C                  = M = Mouse
C
C           nline  = Number of points in line xline,yline.
C
C           iset   = Flag for determining the connection of end-
C                    points (x1,y1,x2,y2) to the line.
C                  = 2 = Connect (x2,y2) to the end of the line
C                        xline(nline),yline(nline).
C                  = 1 = Connect (x1,y1) to the beginning of the
C                        line xline(1),yline(1).
C                  = 0 = Connect (x1,y1) and (x2,y2) as in 1 and
C                        2 above.
C                  =-1 = Do not connect endpoints.
C
C           nptmax = The maximum number of points that nbrpts+
C                    nline can equal.
C
C           nbrpts = Number >=0 used for controlling the number
C                    of points (nline) that may be entered.
C
C-
C ************************************************************************
      Subroutine getlin(xline,yline,nline,x1,y1,x2,y2,iset,nbrpts,
     &nptmax,mcflag,itest)
      Dimension xline(100),yline(100)
      Character mcur*2,mcflag*2
C
      itest=1
      mcur='M'
      If (mcflag.EQ.'M') Then
         Call askmoc(mcur)
         If (mcur.EQ.'Q') Then
```

```
               itest=-1
            Else
               mcflag=mcur
            End If
         Else
            mcur=mcflag
         End If
C
         If (mcur.EQ.'M') Then
            Call linmou(xline,yline,nline,x1,y1,x2,y2,iset,nbrpts,
     &      nptmax,itest)
         Else If (mcur.EQ.'C') Then
            Call lincur(xline,yline,nline,x1,y1,x2,y2,iset,nbrpts,
     &      nptcax,itest)
         End If
C
         Return
         End
C+
C ****************************************************************
C  getmenu - Prompts the user for screen menu item and returns
C            the item number (NUMANS) and the first character of
C            the string associated with the item.
C
C-
C ****************************************************************
         Subroutine getmenu(ans,numans,mcur,xbox0,ybox0,xsize,ysize,
     &xdelbx,ydelbx,nst-ing,string)
         Character*(*) string(nstring),ans*2
C
         If (nstring.GE.1) Then
C
C - Prompt for cursor type, if not already set.
C
            If (mcur.NE.'M'.AND.mcur.NE.'C') Then
               Call askmoc(mcur)
               If (mcur.EQ.'Q') Then
                  itest=-1
                  ans='  '
                  numans=-1
               Else
                  itest=0
               End If
            Else
               itest=0
            End If
C
C - Prompt for screen location.
C
            Do 10 while(itest.EQ.0)
            Call wrtmsg(' Select menu command')
            ierr=1
            Call retpnt(x,y,mcur,ierr)
            If (ierr.GE.0) Then
               itest2=0
               xlftbx=xbox0
               xrgtbx=xbox0+xsize
               ybotbx=ybox0
               ytopbx=ybox0+ysize
               i=1
C
C - Test user entered coordinates against the possible boxes.
C
               Do 20 while(itest2.EQ.0)
```

```
          If ((x.GE.xlftbx.AND.x.LE.xrgtbx).AND.
     &    (y.GE.ybotbx.AND.y.LE.ytopbx)) Then
            ans=string(i)(1:1)//' '
            numans=i
            itest=1
            itest2=1
          Else
            xlftbx=xlftbx+xdelbx
            xrgtbx=xrgtbx+xdelbx
            ybotbx=ybotbx+ydelbx
            ytopbx=ytopbx+ydelbx
            i=i+1
            If (i.GT.nstring) Then
              Call wrtmsg(' Sorry, could not tell which
     & answer you wanted...try again.')
              itest2=-1
            End if
          End If
  20      End Do
C
        Else If (ierr.LT.0) Then
          itest=-1
          ans='  '
          numans=-1
        End If
  10    End Do
C
      Else
        ans='  '
        numans=-1
        Call wrtmsg(' NSTRING passed to GETMENU was <=0')
      End If
C
      Return
      End
C+
C ************************************************************
C  getmou - Returns the world coordinates of the Envision graphics
C           cross-hair cursor when using the mouse.
C-
C ************************************************************
      Subroutine getmou(mode,ix,iy)
C
      Call moulin
      Call mouexm('0')
C
  10  Continue
      Read (5,20,err=90)mode,ix,iy
  20  Format (i3,2z4)
      Call mousop
      Call alphon
      Return
C
  90  Continue
      Call alphon
      Call wrtmsg(' Error, in reading mode and coordinates, try again')
      Go To 10
C
      End
C+
C ***********************************************************************
C  getply - Returns the world coordinates (xscrn,yscrn) of a polygon
C           from the Envision terminal.  The cross-hair cursor keys
C           or the mouse may be used.
```

```
C
C         mcflag = Flag for defaulting to cross-hair cursor type
C                  (mcur takes on the same values as mcflag)
C                = C = Cross-hair cursor keys
C                = M = Mouse
C
C         nbrpts = Number of points in polygon xscrn,yscrn
C-
C ****************************************************************************
      Subroutine getply(xscrn,yscrn,nbrpts,mcflag,itest)
      Dimension xscrn(100),yscrn(100)
      Character mcur*2,mcflag*2
C
      nbrpts=0
      itest=1
      mcur='N'
      If (mcflag.EQ.'N') Then
         Call askmoc(mcur)
         If (mcur.EQ.'Q') Then
            mcflag='N'
            itest=-1
         Else
            mcflag=mcur
         End If
      Else
         mcur=mcflag
      End If
C
      If (mcur.EQ.'M') Then
         Call entmou(xscrn,yscrn,nbrpts,itest)
      Else If (mcur.EQ.'C') Then
         Call entply(xscrn,yscrn,nbrpts,itest)
      End If
C
C - Test nbrpts against itest...(did user crash out or what)
C   also test nbrpts less than or equal to zero...
C
      Return
      End
C+
C *****************************************************************
C  getsub - Gets the subgrid locations entered using the cursor
C           keys or Envision mouse.
C-
C *****************************************************************
      Subroutine getsub(itest)
      Common /subscreen/xscrn(2),yscrn(2),xgrid(2),ygrid(2)
      Character quest*80,mcflag*2,mcur*2,ans*1,id*56,pgm*8,lntype*1
      Character votflg*2
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /gridspecs/id,pgm,nc,nr,xo,dx,yo,dy,iproj,cm,bl
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /flags/mcflag,votflg
C
      itest=0
      If (mcflag.EQ.'N') Then
         Call askmoc(mcur)
         If (mcur.EQ.'Q') Then
            itest=-1
            Else
            mcflag=mcur
            End If
         Else
         mcur=mcflag
         End If
      If (itest.EQ.-1)Go To 100
C
C - Print help message
C
      If (votflg.EQ.'V') Then
         Call hlpmou
         Call wait
         End If
C
      itest2=0
      i=0
      Do 10 while(itest2.EQ.0)
         Call retpnt(x,y,mcur,iret)
         If (iret.GE.1) Then
            i=i+1
            If (i.EQ.1) Then
               xscrn(1)=x
               yscrn(1)=y
            Else
               xscrn(2)=x
               yscrn(2)=y
               itest2=1
            End If
         Else
            itest2=-1
            itest=-1
         End If
   10 End Do
      If (itest2.EQ.-1)Go To 100
C
      lntype='0'
      Call setlin(lntype)
      Call drwlin(xscrn(1),yscrn(1),xscrn(1),yscrn(2))
      Call drwlin(xscrn(1),yscrn(2),xscrn(2),yscrn(2))
      Call drwlin(xscrn(2),yscrn(2),xscrn(2),yscrn(1))
      Call drwlin(xscrn(2),yscrn(1),xscrn(1),yscrn(1))
C
      xsc=float(nxpix)
      ysc=float(nypix)
C
      If (xscrn(1).GT.xscrn(2)) Then
         xdum=xscrn(1)
         xscrn(1)=xscrn(2)
         xscrn(2)=xdum
         End If
C
      If (yscrn(1).GT.yscrn(2)) Then
         ydum=yscrn(1)
         yscrn(1)=yscrn(2)
         yscrn(2)=ydum
         End If
C
      rscxsc=0.0
      rscysc=0.0
      If (abs(xsc).GT.1.0e-16)rscxsc=1.0/xsc
      If (abs(ysc).GT.1.0e-16)rscysc=1.0/ysc
      Do 20 j=1,2
      xgrid(j)=xstart+(xscrn(j)-iwcorg)*rscxsc
      ygrid(j)=ystart+(yscrn(j)-jwcorg)*rscysc
   20 Continue
C
```

```
      ncmin=int(xgrid(1))
      ncmax=int(xgrid(2)+0.5)
      nrmin=int(ygrid(1))
      nrmax=int(ygrid(2)+0.5)
C
      If (ncmin.LT.1)ncmin=1
      If (ncmax.GT.nc)ncmax=nc
      If (nrmin.LT.1)nrmin=1
      If (nrmax.GT.nr)nrmax=nr
C
      itest=1
      Call zomstr(itest)
C
  100 Continue
      Return
      End
C+
C **************************************************************
C  graphoff - Turns off the graphics character mode.  See page 5-8
C             in Envision Reference Manual, May 23, 1983.
C-
C **************************************************************
      Subroutine graphoff
      Character com*3
C
      com='RG0'
      Call esccom(com)
C
      Return
      End
C+
C **************************************************************
C  grdin  - Asks the user whether a previous grid should be used
C           or if a new grid should be created.
C-
C **************************************************************
      Subroutine grdin(itest)
      Character quest*80,ans*2
C
      itest=0
      Do 10 while(itest.EQ.0)
         quest=' Read or create a grid (r/c/q)'
         ival=iaquest(quest,ans,'(a2)',0)
C
         If (ans.EQ.'R') Then
            Call getgrd(itest)
         Else If (ans.EQ.'C') Then
            Call crtgrd(itest)
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
         If ((ans.EQ.'R'.OR.ans.EQ.'C').AND.itest.EQ.0)itest=1
   10 End Do
C
      Return
      End
C+
C **************************************************************
C  grvmod - Writes out a GRAVPOLY model using the POLYGON model
C           information and user entered responses.
C-
C **************************************************************
```

```
      Subroutine grvmod(itest)
      Dimension xpoly(100),ypoly(100)
      Common /topology/info(100),iupper(100),idown(100),ileft(100),
     &iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /parameter /parm(100,10)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Character idplot*40,ifile*50,ifile2*50,quest*80,ifmt*80,
     &id*56,pgm*8,confil*80,ans*2,ans2*2,name*8,unix*4,uniz*4
      namelist/parms/iplotr,ibody,ifile,ifile2,lsqs,datum,xscale,
     &dc,unix,uniz,name,height,ifmt,idplot,iobs,icalc,ires,naxcol
C
      iplotr=9
      nuxcol=130
      ibody=0
      iobs=0
      icalc=0
      ires=0
      idplot=' '
      ifile=' '
      ifile2=' '
      lsqs=0
      datum=0.
      xscale=0.
      dc=0.
      unix='kilm'
      uniz='feet'
      name='gridded'
      height=0.
C
C - Ask for name of file containing fieldpoint information.
C
      itest=1
      quest=' Name of file containing fieldpoint information'
      ival=iaquest(quest,ifile,'(a50)',0)
      If (ival.EQ.-1)Go To 100
C
C - Name (type of data read in from ifile).
C
      itest=0
      Do 40 while(itest.EQ.0)
         quest=' Type of data in fieldpoint file (enter H for help)'
         ival=iaquest(quest,name,'(a8)',-5)
C
         If (name(1:1).EQ.'G') Then
            name='gridded'
            height=0.0
            Write (6,45)
   45 Format (2x,'Height of fieldpoint grid above the same datume that')
            quest=' is used to reference the body heights'
            ival=irquest(quest,height,'(e16.8)',16)
            itest=1
            If (ival.EQ.-1)itest=-1
         Else If (name.EQ.'H') Then
            Call hlpg2
         Else If (ival.EQ.-1) Then
            itest=-1
         Else
C
C - Ask for format type of data.
C
            itest2=0
            Do 25 while(itest2.EQ.0)
```

```
               quest=' Enter format to use when reading in your file'
               ival=iaquest(quest,ifmt,'(a80)',0)
               If (ival.EQ.-1) Then
                  itest=-1
                  itest2=-1
               Else If (ifmt.NE.' ') Then
                  itest2=1
                  itest=1
               Else
                  Call errmsg
               End If
   25       End Do
         End If
   40 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for the name of standard grid containing the heights of
C   fieldpoints.
C
      quest=' Name of standard grid containing heights of fieldpoints'
      ival=iaquest(quest,ifile2,'(a50)',0)
      If (ival.EQ.-1)Go To 100
C
C - Ask for print out identifier.
C
      quest=' Identifier for printer output'
      ival=iaquest(quest,idplot,'(a40)',0)
      If (ival.EQ.-1)Go To 100
C
C - Ask if constant should be added to calculated anomalies.
C
      itest=0
      Do 20 while(itest.EQ.0)
         quest=' Add a constant to the calculated anomalies (y/n/q)'
         ival=iaquest(quest,ans,'(a2)',0)
C
         If (ans.EQ.'Y') Then
            quest=' Constant to be added'
            ival=irquest(quest,datum,'(e16.8)',16)
            itest=1
            If (ival.EQ.-1)itest=-1
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else If (ans.EQ.'N') Then
            itest=1
            datum=0.0
         Else
            Call errmsg
         End If
   20 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask lsqs determination.
C
      itest=0
      Do 30 while(itest.EQ.0)
         quest=' Least-squares comparison/read observed values
     & (y/n/h/q)'
         ival=iaquest(quest,ans2,'(a2)',0)
C
         If (ans2.EQ.'Y') Then
            itest2=0
            Do 35 while(itest2.EQ.0)
               lsqs=0
```

```
              quest=' What value for LSQS (0/1/7, // to quit)'
              ival=iiquest(quest,lsqs,'(i2)',-2)
C
              If (lsqs.EQ.0.OR.lsqs.EQ.1.OR.lsqs.EQ.7) Then
                 itest2=1
                 itest=1
              Else If (ival.EQ.-1) Then
                 itest2=-1
                 itest=-1
              Else
                 Call errmsg
              End If
   35       End Do
        Else If (ans2.EQ.'N') Then
           lsqs=0
           itest=1
        Else If (ans2.EQ.'H') Then
           Call hlpg1
        Else If (ans2.EQ.'Q'.OR.ival.EQ.-1) Then
           itest=-1
        Else
           Call errmsg
        End If
   30 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for units in x&y directions.
C
      itest=0
      Do 50 while(itest.EQ.0)
         unix='KILM'
         quest=' Units in x&y direction (FEET/KILF/MILE/METR/KILM/H/Q)'
         ival=iaquest(quest,unix,'(a4)',-4)
C
         itest=1
         If (unix.EQ.'FEET') Then
            unix='feet'
         Else If (unix.EQ.'MILE') Then
            unix='mile'
         Else If (unix.EQ.'KILF') Then
            unix='kilf'
         Else If (unix.EQ.'METR') Then
            unix='metr'
         Else If (unix.EQ.'KILM') Then
            unix='kilm'
         Else If (unix.EQ.'H') Then
            Call hlpg3
            itest=0
         Else If (unix.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
            itest=0
         End If
   50 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for units in z direction (height).
C
      itest=0
      Do 60 while(itest.EQ.0)
         uniz='FEET'
         quest=' Units in z direction (FEET/KILF/MILE/METR/KILM/H/Q)'
         ival=iaquest(quest,uniz,'(a4)',-4)
```

```
C
          itest=1
          If (uniz.EQ.'FEET') Then
             uniz='feet'
          Else If (uniz.EQ.'MILE') Then
             uniz='mile'
          Else If (uniz.EQ.'KILF') Then
             uniz='kilf'
          Else If (uniz.EQ.'METR') Then
             uniz='metr'
          Else If (uniz.EQ.'KILM') Then
             uniz='kilm'
          Else If (uniz.EQ.'H') Then
             Call hlpg3
             itest=0
          Else If (uniz.EQ.'Q'.OR.ival.EQ.-1) Then
             itest=-1
          Else
             Call errmsg
             itest=0
          End If
   60 End Do
       If (itest.EQ.-1)Go To 100
C
C - Ask for print switch to use.
C
       itest=0
       Do 65 while(itest.EQ.0)
          iflag=0
          quest=' Print switch for body information (1/0)'
          ival=iiquest(quest,iflag,'(i1)',1)
C
          If (ival.EQ.-1) Then
             itest=-1
          Else If (iflag.EQ.0.OR.iflag.EQ.1) Then
             itest=1
          Else
             Call errmsg
          End If
   65 End Do
       If (itest.LT.0)Go To 100
C
C - Ask which parm from the parm arrays to use when creating
C   this model.
C
       Call askgvl(ione,itwo,ithree,itest)
       If (itest.EQ.-1)Go To 100
C
C - Ask for the name to call this GRAVPOLY file (comfil)
C
   70 Continue
       quest=' Name to call command file to run GRAVPOLY'
       ival=iaquest(quest,comfil,'(a80)',0)
       If (ival.EQ.-1)Go To 100
       If (comfil.EQ.' ')Go To 70
C
C - Open the model file and write out the model.
C
       Open (11,file=comfil,status='new',carriagecontrol='list')
       Write (11,nml=parms)
       Call fndtop(ntop)
       next=ntop
       itest=0
       delxcn=dx/nxorgp
```

```
      delycn=dy/nyorgp
      Do 150 while(itest.EQ.0)
         ngon=next
         npoly=info(ngon)
         nbrpts=numply(npoly)
         If (nbrpts.GT.0) Then
            Do 120 i=1,nbrpts
            xpoly(i)=xo+(xscr(npoly,i)-iwcorg)*delxcn
            ypoly(i)=yo+(yscr(npoly,i)-jwcorg)*delycn
  120       Continue
C
C - Find the handedness of the polygon (by right hand rule).
C      gamma > 0.0  = polygon is counter-clockwise
C      gamma < 0.0  = polygon is clockwise
C
            Call totalt(gamma,xpoly,ypoly,nbrpts)
C
            If (gamma.GT.0) Then
               nstart=nbrpts
               nend=1
               istep=-1
            Else
               nstart=1
               nend=nbrpts
               istep=1
            End If
C
C - If the body has a parent, we need to determine if the parent-
C   body/current-body overlap.
C
            iup=iupper(ngon)
            If (iup.GT.0) Then
               nplyup=info(iup)
C
               partop=parm(nplyup,ione)
               parbot=parm(nplyup,itwo)
               bodtop=parm(npoly,ione)
               bodbot=parm(npoly,itwo)
C
C - We need only remove a slab if the parent body and current body
C   overlap in the z direction (height).
C
               If (((bodtop.LE.partop).AND.(bodbot.GE.parbot)).OR.
     &         ((bodtop.GT.partop).AND.(bodbot.LT.partop)).OR.
     &         ((bodtop.GT.parbot).AND.(bodbot.LT.parbot))) Then
C
                  If (bodtop.LE.partop) Then
                     ztop=amin1(bodtop,partop)
                  Else
                     ztop=partop
                  End If
C
                  If (bodbot.GE.parbot) Then
                     zbot=amax1(bodbot,parbot)
                  Else
                     zbot=parbot
                  End If
C
C - Write out the slab parameters and coordinate information.
C
                  Write (11,*)nbrpts,iflag,-parm(nplyup,ithree),
     &         ztop,zbot
                  Write (11,*)(xpoly(i),ypoly(i),i=nstart,nend,istep)
               End If
            End If
C
C - Write out the parameters and current body coordinate info.
C
            Write (11,*)nbrpts,iflag,parm(npoly,ithree),
     &      parm(npoly,ione),parm(npoly,itwo)
            Write (11,*)(xpoly(ii),ypoly(ii),ii=nstart,nend,istep)
         End If
C
C - Get next polygon
C
         Call walk(next,ngon,itest2)
         If (next.EQ.0.OR.itest2.LE.0)itest=1
  150 End Do
      Close (11)
C
  100 Continue
      Return
      End
C+
C **************************************************************
C  gtpnt  - Returns the current world coordinates (x,y) of the
C           Envision cross-hair cursor.
C
C          ans = Character answer that user entered [format(a1)]
C              = e or f, prompts the terminal for screen coord.,
C                and returns character answer.
C              = h = returns ans='h'
C              = q = returns ans='q'
C-
C **************************************************************
      Subroutine gtpnt(ans,x,y)
      Character dum*3,ans*1
C
      idx=0
      idy=0
      Call setkam
      Call loccur(ix,iy)
      itest=0
      If (itest.EQ.0) Then
   10    Continue
         dum='   '
         Read (5,15,err=10)dum
   15    Format (a3)
   20    Continue
C
         If (dum.EQ.'E'.OR.dum.EQ.'e') Then
            Call loccur(ixloc,iyloc)
            x=float(ixloc)
            y=float(iyloc)
            ans='e'
            itest=1
C
         Else If (dum.EQ.'F'.OR.dum.EQ.'f') Then
            Call loccur(ixloc,iyloc)
            x=float(ixloc)
            y=float(iyloc)
            ans='f'
            itest=1
C
         Else If (dum.EQ.'H'.OR.dum.EQ.'h') Then
            ans='h'
            itest=1
C
```

```
         Else If (dum.EQ.'Q'.OR.dum.EQ.'q') Then
            ans='q'
            itest=1
C
         Else If (dum.EQ.'OS') Then
            idx=1
            idy=1
C
         Else If (dum.EQ.'Om') Then
            idx=-1
            idy=1
C
         Else If (dum.EQ.'Ol') Then
            idx=-1
            idy=-1
C
         Else If (dum.EQ.'Ox') Then
            idy=1
C
         Else If (dum.EQ.'Ot') Then
            idx=-1
C
         Else If (dum.EQ.'Or') Then
            idy=-1
C
         Else If (dum.EQ.'Ov') Then
            idx=1
C
         Else If (dum.EQ.'[\"') Then
            idy=10
C
         Else If (dum.EQ.'[\$') Then
            idx=-10
C
         Else If (dum.EQ.'[\(') Then
            idy=-10
C
         Else If (dum.EQ.'[\&') Then
            idx=10
C
         End If
C
         If (itest.EQ.0) Then
            ix=ix+idx
            iy=iy+idy
            Call movcur(ix,iy)
            idx=0
            idy=0
            Go To 10
         End If
      End If
C
      Return
      End
      subroutine hlpadd(itest)
c
      itest=1
      print *,' Help for adding polygons not available'
c
      return
      end
C+
C ****************************************************************
C  hlpapt - Help message for subroutine addpnt (ADD POINT).
```

```
C-
C **************************************************************
      Subroutine hlpapt
      Character ans*1
C
      Write (6,10)
   10 Format (/,' Position the cross-hair cursor to a corner of',
     &' a polygon, then choose:',/,
     &/,'        (KEY)  (MOUSE)',
     &/,'       e or f  1 or 2 = Selects the polygon',
     &/,'         h        3   = Help message',
     &/,'         q       2&3  = Quit and return to Edit polygon mode',/)
C
      Write (6,20)
   20 Format (' When the polygon you selected is found, it will',
     &' be drawn in white on the',/,' screen and you will be asked',
     &' to select (using the cross-hair cursor):',/,
     &/,'      (1) The side of the polygon where points will be added',
     &/,'      (2) An endpoint of the selected side.',/)
C
      Call wait
C
      Return
      End
C+
C **************************************************************
C  hlpchg - HeLP CHanGe - Displays the options available for
C           change labels/parameters associated with polygons.
C-
C **************************************************************
      Subroutine hlpchg
C
      Write (6,10)
   10 Format (/,'  Change mode options:',/
     &/,'           l = Change/assign labels on parameter list',
     &/,'           p = Change/assign parameters to polygons',
     &/,'           q = Quit and return to Polygon add/.../edit mode',/)
C
      Return
      End
C+
C ***************************************************************
C  hlpcom - Help module for Polygon command level, called by
C           the program POLYGON.
C-
C **************************************************************
      Subroutine hlpcom
C
      Write (6,10)
   10 Format (/,' Polygon options available:',//,
     &'       p = Polygon (add/change_parm/delete/edit_poly) mode',/
     &'       o = Output Plouff/Godson file or standard grid',/,
     &'       r = Read model from file',/,
     &'       w = Write model to output file',/,
C    & '       $ = Command parameter change mode',/,
     &'       z = Zoom to a subgrid of grid',/,
     &'       Q = quit',/)
C
      Return
      End
C+
C ***************************************************************
C  hlpcpl - Help message for subroutine pckpnt (PICK POINT)
C           for picking with the cross-hair cursor the corner of
```

```
C          a polygon.
C-
C *************************************************************
      Subroutine hlpcpl
C
      Write (6,10)
   10 Format (/,' Position the cross-hair cursor to a corner of',
     &/,' this polygon and enter the screen location.',/)
C
      Return
      End
C+
C *************************************************************
C  hlpcpy - Prints help message about using the CPYPLY (COPY POLYGON)
C          feature.
C-
C *************************************************************
      Subroutine hlpcpy
C
      Write (6,10)
   10 Format (/,'  To copy a polygon:',/,
     &/,'    (i)  Position graphics cursor to a corner of polygon to',
     &' copy',
     &/,'   (ii)  Use the following keys to identify the polygon',
     &' or screen location',
     &//,'         MOUSE   KEYBOARD   FUNCTION',
     &/,'           1        "e"      Sends cursor position to program',
     &/,'           2        "f"        "      "      "     "     "',
     &/,'           3        "h"      Help',
     &/,'          2&3       "q"      Quit',/,
     &/,' (iii)  Position the graphics cursor to the location to copy',
     &' the polygon at',
     &/,'   (iv)  Press appropriate key(s) as in step (ii).',
     &//,'  NOTE:  The parameters for this polygon are also copied.',/)
C
      Return
      End
C+
C *************************************************************
C  hlpcur - Help message for entering a single screen location
C          using the cross-hair cursor keys and keyboard com-
C          mand keys.  Called by subroutine retpnt (RETURN POINT)
C-
C *************************************************************
      Subroutine hlpcur
C
      Write (6,10)
   10 Format (/,' To enter a screen location, position the cross-hair',
     &' cursor to the desired',/,' location and then type:',/,
     &/,'    e or f = Enters the cross-hair cursor location',
     &/,'        h   = Help',
     &/,'        q   = Quit (no point is entered)',/)
C
      Return
      End
C+
C *************************************************************
C  hlpdel - Prints help message on using the delply (DELETE POLY-
C          GON) mode.
C-
C *************************************************************
      Subroutine hlpdel
C
      Print *,' Help for deleting polygon not available yet'
```

```
C
      Return
      End
C+
C *************************************************************
C  hlpdpt - Help message for subroutine delpnt (DELETE POINT).
C-
C *************************************************************
      Subroutine hlpdpt
      Character ans*1
C
      Write (6,10)
   10 Format (/,' Position the cross-hair cursor to a corner of',
     &' a polygon, then type:',/,
     &/,'      (KEY)  (MOUSE)',
     &/,'      e or f  1 or 2 = Selects the polygon',
     &/,'        h       3    = Help message',
     &/,'        q      2&3   = Quit and return to Edit polygon',
     &' mode',/)
C
      Write (6,20)
   20 Format (' When the polygon you selected is found, it will',
     &' be drawn in white on the screen',/,' and you will be asked',
     &' to select (using cross-hair cursor), the corner of this',
     &/,' polygon to delete.',/)
C
      Call wait
C
      Return
      End
C+
C *************************************************************
C  hlpedt - Help message for Edit polygon mode.
C-
C *************************************************************
      Subroutine hlpedt
C
      Write (6,10)
   10 Format (/,'  Edit mode options:',/,
     &/,'         a = Add points',
     &/,'         d = Delete a point',
     &/,'         m = Move a point',
     &/,'         q = Quit and return to Polygon mode',/)
C
      Return
      End
C+
C **************************************************************
C  hlpent - Asks if help is need for entering a polygon.
C-
C **************************************************************
      Subroutine hlpent(itest)
      Character ans*1,answ*1,quest*80
C
      itest=0
      If (itest.EQ.0) Then
   10    Continue
         answ='y'
         quest=' Do you need instructions (y/n)'
         ival=iaquest(quest,answ,'(a1)',1)
C
         If (answ.EQ.'y') Then
            Call msgent
            itest=1
         Else If (answ.EQ.'n') Then
            itest=1
         Else If (answ.EQ.'/') Then
            itest=-1
         Else
            Call errmsg
         End If
         If (itest.EQ.0)Go To 10
      End If
C
      Return
      End
C+
C **************************************************************
C  hlpg1 - Prints help message at user's terminal about least-
C          squares determination when generating a GRAVPOLY file.
C-
C **************************************************************
      Subroutine hlpg1
C
      Write (6,10)
   10 Format (/,'  LSQS - A number that determines if fieldpoint',
     &' (observed) values will be',/,' read or not or whether a',
     &' least-squares comparison will be made between the',/,
     &' calculated and observed values.',//,
     &'        0 = No fieldpoint (observed) values will be used.',
     &' It must be kept in mind',/,'            that it is always',
     &' necessary to read fieldpoint locations in order to',
     &/,'            define the grid points at which the model',
     &' values will be calculated.',
     &/,'        1 = Fieldpoint values will be read and used to'
     &' calculate residual',/,'            values (observed-',
     &'calculated).',
     &/,'        7 = A least-squares comparison between the observed',
     &' and calculated values',/,'            will be made to',
     &' determine the best density contrast.',/)
C
      Return
      End
C+
C **************************************************************
C  hlpg2 - Prints help message at user's terminal about data type
C          of fieldpoint file.
C-
C **************************************************************
      Subroutine hlpg2
C
      Write (6,10)
   10 Format (/,'  Enter the type of data in the fieldpoint file.',
     &'  Default is "gridded" or "g",',/,' which means that the',
     &' fieldpoint data will be read as a standard USGS geophysics',
     &' gridded file.',/,' Any other character (maximum of 8) will',
     &' be assusmed a user formatted file.',/,' If a user formatted',
     &' file is read, a standard USGS geophysics file of the same',/,
     &' data with the name "GRAVPOLY.OBS" will be created.',/)
C
      Return
      End
      subroutine hlpg3
c
      print *,' Made it to hlpg3'
c
      return
      end
```

```
C           squares determination when generating a MAGPOLY file.
C-
C **************************************************************
      Subroutine hlpmgl
C
      Write (6,10)
   10 Format (/,
     &'  LSQS is a number that determines what type of field-',
     &'point (observed) values',/,' will be read or what type or',
     &' least-squares comparison will bu made between',/,' the',
     &' calculated and observed data.',
     &/,'         0 = No fieldpoint (observed) values will be used.',
     &'  It must be kept',/,
     &'             in mind that it is always necessary to read',
     &' field point',/,
     &'             locations in order to define the grid points',
     &' at which the model',/,
     &'               values will be calculated',/
     &/,'         1 = y component of magnetic anomaly',
     &/,'         2 = x component of magnetic anomaly',
     &/,'         3 = Vertical component of magnetic anomaly (some',
     &' ground surveys)',
     &/,'         4 = Total horizontal component of magnetic anomaly',
     &/,'         5 = Total magnetic anomaly (the usual observed',
     &' aeromagnetic anomaly)',
     &/,'         6 = A best total magnetization vector (amplitude,',
     &' declination and',/,
     &'             inclination) will be determined by a least-',
     &'square comparison',
     &/,'             between the observed and calculated values.',
     &/,'         7 = A best susceptibility, assuming no remanent',
     &' magnetization, w:ll',
     &/,'             be determined by a least-squares comparison',
     &' between the',
     &/,'             observed and calculated values.',/)
C
      Return
      End
C+
C *********************************************************************
C  hlpmou - Prints help messages on users terminal about using
C           the Envision mouse or cross-hair cursor keys.
C-
C *********************************************************************
      Subroutine hlpmou
C
      Call msgmo1
      Call msgmo2
      Call msgmo3
C
      Return
      End
C+
C **************************************************************
C  hlpmov - Prints help message about using the MOVPLY (MOVE POLYGON)
C           feature.
C-
C **************************************************************
      Subroutine hlpmov
C
      Write (6,10)



     Return
```

```
      End
C+
C *************************************************************
C  hlpmvp - Help message for moving a single point, called
C           by subroutine movpnt (MOVE POINT).
C-
C *************************************************************
      Subroutine hlpmvp
      Character ans*1
C
      Write (6,10)
   10 Format (/,' To move a corner point, position the cross-hair',
     &' cursor to a corner of the',/,' polygon, then type:',/,
     &/,'    (KEY)   (MOUSE)',
     &/,'    e or f  1 or 2 = Enters the cross-hair cursor location',
     &/,'       h       3    = Help',
     &/,'       q      2&3   = Quit (no point is entered)',/)
C
      Write (6,20)
   20 Format (' The polygon that you selected will be drawn in white',
     &' on the screen.  The point',/,' to be moved is entered',
     &' by positioning the cross-hair cursor over the desired',
     &/,' corner of the polygon (entering this corner location),',
     &' and then positioning the',/,' cursor to the new location',
     &' for this corner.',/)
C
      Call wait
C
      Return
      End
C+
C *************************************************************
C  hlpout - HeLP OUT - Prints help message at user's terminal
C           about Polygon output file types.
C-
C *************************************************************
      Subroutine hlpout
C
      Write (6,10)
   10 Format (/,'  Output file options:',/
     &/,'           p = Plouff/Godson format file (GRAVPOLY/MAGPOLY)',
     &/,'           s = Standard grid file',
     &/,'           q = Quit and return to POLYGON command',/)
C
      Return
      End
C+
C *************************************************************
C  hlpvaj - Prints help message about options available in the
C           chgpar (CHANGE PARAMETER) mode.
C-
C *************************************************************
      Subroutine hlppar
C
      Write (6,10)
   10 Format (/,'   Options for changing polygon parameters:',/,
     &/,'           a = All polygons (via tree structure)',
     &/,'           p = Pick individual polygons using cursor',
     &/,'           q = Quit',/)
C
      Return
      End
C+
C **************************************************************
```

```
 hlpplf - Help message for PLOUFF/GODSON GRAVPOLY/MAGPOLY out-
          put files.

******************************************************************
     Subroutine hlpplf

     Write (6,10)
  10 Format (/,' Enter "g" for a gravity model, or "m" for a magnetic'
    &,
    &' model.',
    &/,' NOTE: GRAVPOLY and MAGPOLY assume that the positive',
    &' direction is upward.',/)

     Return
     End

******************************************************************
 hlpply - Prints help message for plycom (POLYGON COMMAND) sub-
          routine.

******************************************************************
     Subroutine hlpply

     Write (6,10)
  10 Format (/,'  POLYGON add/change/delete/edit options:',/
    &/,'            a = Add a polygon',
    &/,'            c = Change polygon parameters',
    &/,'            d = Delete a polygon',
    &/,'            e = Edit a polygon',
    &/,'            s = Special functions',
    &/,'            q = Quit and return to POLYGON command',/)

     Return
     End

******************************************************************
 hlprot - Prints help message about using the ROTPLY (ROTATE POLYGON)
          feature.

******************************************************************
     Subroutine hlprot

     Write (6,10)
  10 Format (/,'  To rotate a polygon:',/,
    &/,'     (i)  Position the graphics cursor to a corner of',
    &' the polygon to rotate',
    &/,'    (ii)  Use the following keys to identify the polygon',
    &' or screen location',
    &//,'         MOUSE    KEYBOARD   FUNCTION',
    &/,'           1         "e"     Sends cursor position to program',
    &/,'           2         "f"      "      "        "    "    "',
    &/,'           3         "h"     Help',
    &/,'          2&3        "q"     Quit',/,
    &/,'   (iii)  Choose the location to use for the center of',
    &' rotation',
    &/,'          (corner picked in steps (i) and (ii) may be used',
    &/,'    (iv)  Press appropriate key(s) as in step (ii).',
    &/,'     (v)  Enter rotation angle in degrees.',/)

     Return
     End

******************************************************************
 hlprt2 - Help for subroutine rotply (ROTATE POLYGON), sends
```

```
C           message to user on positioning cursor for origin to
C           rotate polygon about.
C-
C **************************************************************
      Subroutine hlprt2
C
      Write (6,10)
   10 Format (/,' Move the cross-hairs to the location of the origin',
     &' to rotate this',/,' polygon about, and enter this point.',/)
C
      Return
      End
C+
C **************************************************************
C  hlpspf - Help information for SPFCOM - (SPECIAL FUNCTIONS MODE)
C-
C **************************************************************
      Subroutine hlpspf
C
      Write (6,10)
   10 Format (/,' Special functions are:',/,
     &/,'        c = Copy a polygon,',
     &/,'        m = Move a polygon,',
     &/,'        r = Rotate a polygon,',
     &/,'        q = Quit.',/)
C
      Return
      End
C+
C **************************************************************
C  hlpstk - Help message for stack release mode.  Called by sub-
C           routine zomstk (ZOOM STACK).
C-
C **************************************************************
      Subroutine hlpstk
C
      Write (6,10)
   10 Format (/,'  Zoor stack release options:',/,
     &/,'       b = bottom of stack (oldest zoom values)',
     &/,'       c = clear zoom stack',
     &/,'       t = top of stack (youngest zoom values)',
     &/,'       q = quit or // return to zoom command level',/)
C
      Return
      End
C+
C **************************************************************
C  hlpzom - Help message for zoom command mode.
C-
C **************************************************************
      Subroutine hlpzom
C
      Write (6,10)
   10 Format (/,' Zoom options available:',/,
     &/,'       c = Clear zoom stack',
     &/,'       d = Draw grid using a selected zoom value',
     &/,'       r = Recall and draw grid using zoom values at top of',
     &' stack',
     &/,'       s = Select a subgrid (using mouse, cursor, or coords)',
     &/,'       u = Unzoom and draw grid on screen (does nothing',
     &/,'           to zoom values)',
     &/,'       q = Quit and return to Polygon command',/)
C
      Return
      End
C+
C    IAQUEST -asks question with character answer
C
C     ireturn = IAQUEST (quest,aval,form,mode)
C
C      ireturn =  -1 if '//' was given as response (user wants out)
C                  0 if no response (user took default)
C                  1 if user responded (returns response in aval)
C
C        quest =  Character string containing question to be asked
C                 (with no ? at the end, it is added by function)
C
C         aval =  Character string to receive answer
C                 (used to pass default if one is available)
C
C         form =  Character string containing fortran format to be
C                 used to read the user response
C
C         mode =  Integer control parameter:
C
C              mode > 0, default allowed
C              mode = 0, no default allowed, upshift response
C              mode < 0, default allowed, upshift response
C
C-
C  ***********************************************
      Integer Function iaquest(quest,aval,form,mode)
      Character quest*(*),form*(*),aval*(*)
      Character*100 str,form2*10,ans*80,astr*30
      iaquest=0
      iqlen=itlen(quest)
      irlen=itlen(aval)
      If (mode.LT.0)Call upshift(aval)
      If (irlen.EQ.0)irlen=1
      If (mode.NE.0) Then
         str=quest(1:iqlen)//' ['//aval(1:irlen)//']?'
         islen=iqlen+irlen+4
         Else
         str=quest(1:iqlen)//'?'
         islen=iqlen+1
         End If
      Write (form2,105)islen
  105 Format ('(x,a',i3,',$)')
   13 Write (6,form2)s':
      Read (5,form)ans
      ialen=31
      ialen=itlen(ans)
      If (ialen.NE.0) Then
         If (ans.EQ.'//') Then
            iaquest=-1
            Else
            iaquest=1
            aval(1:len(aval))=ans(1:len(aval))
            If (mode.LE.0)Call upshift(aval)
            End If
         Else
         If (mode.EQ.0)Go To 13
         End If
      Return
      End
C+
C    IDEBLANK- removes all blanks and returns length
C
```

```
C     ireturn = IDEBLANK (string)
C
C        ireturn = length of the string without blanks
C         string = character string to be de-blanked
C
C-
C  **********************************************
      Integer Function ideblank(string)
      Character*256 string*(*),temp
      j=1
      temp=' '
      itemp=len(string)
      Do 10 i=1,itemp
      If (string(i:i).EQ.' ')Go To 10
      temp(j:j)=string(i:i)
      j=j+1
   10 Continue
      string=temp
      ideblank=j-1
      Return
      End
C+
C     IIQUEST - Asks a question with an integer answer
C
C      ireturn = IIQUEST (quest,ival,form,mode)
C
C       ireturn =  -1 if '//' was given as response (user wants out)
C                   0 if no response (user took default)
C                   1 if user responded (returns response in aval)
C
C         quest =  Character string containing question to be asked
C                  (with no ? at the end, it is added by function)
C
C          ival =  integer variable to receive answer
C                  (used to pass default if one is available)
C
C          form =  Character string containing fortran format to be
C                  used to format the default contained in ival
C
C          mode =  Integer control parameter:
C
C             mode = 0, required response (no default allowed)
C             mode <>0, default allowed
C-
C  **********************************************
      Integer Function iiquest(quest,ival,form,mode)
      Character quest*(*),form*(*),rstr2*30
      Character*100 str,form2*10,ans*30,astr*30
      iiquest=0
      iqlen=itlen(quest)
      Write (rstr2,form)ival
      irlen=ideblank(rstr2)
      If (mode.NE.0) Then
         str=quest(1:iqlen)//' ['//rstr2(1:irlen)//']?'
         islen=iqlen+irlen+4
         Else
         str=quest(1:iqlen)//'?'
         islen=iqlen+1
         End If
      Write (form2,105)islen
  105 Format ('(x,a',i3,',$)')
   13 Write (6,form2)str
      Read (5,110)ans
  110 Format (a30)
```

```
      ialen=ideblank(ans)
      If (ialen.NE.0) Then
         If (ans.EQ.'//') Then
            iiquest=-1
            Else
            iiquest=1
            Read (ans(1:ialen),120,err=25)ival
  120       Format (I10)
            End If
         Else
         If (mode.EQ.0)Go To 13
         End If
      Return
C     couldn't decode
   25 Continue
      Write (6,130)
  130 Format (' Please answer again, I expect a number.')
      Go To 13
      End
C+
C ***************************************************************
C  initial - Initializes the variables used in running POLYGON.
C-
C ***************************************************************
      Subroutine initial
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /misc/ncol,nrow,first,ntop,ifirst
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Character id*56,pgm*8
C
C - Initialize the maximum number of polygons (nmax), the maximum
C   number of points in each polygon (nptmax), and the starting
C   number of polygons (ntotal).
C
      nmax=100
      nptmax=100
      ntotal=0
C
C - Initialize variables used in drawing the grid and polygons
C   on the terminal screen.
C
      Call intscr
C
C - Initialize the variables stored in the original common block
C
      Call intorg
C
C - Initializes the calc common block
C
      Call intcal
C
C - Initialize flags used in program
C
      Call intflg
C
C - Initialize the zoom stack and pointers.
C
      izoom=0
      izval=0
      Call intzom
C
C - Initialize the location arrays for the polygons.
```

```
C
      Call intloc
C
C - Initialized the topology structure arrays.
C
      Call inttpl
      Call intnew
C
C - Initialize the labels associated with the value list.
C
      Call intlab
C
C - Initialize the parameters in the perm list.
C
      Call intpar
C
C - Initialize the paramters used in filling a polygon
C
      Call intfil
C
      Return
      End
C+
C ***************************************************************************
C  inkjet - Sets an optimal color map for dumping the screen to the
C           inkjet plotter.  At present, only ncolors=12 properly map.
C
C           Author:  Robert W. Simpson, USGS, Menlo Park, CA, 1984.
C-
C ****************************************************************************
      Subroutine inkjet(iounit,ncolors)
C
      Character*100 str(20)
      If (ncolors.GE.1.AND.ncolors.LE.13) Then
C
         str(13)='1J0007,1000,3165,4110,5245,6006,7090,8210,9003,'//
     &   ':221,;107,<075,=077,>111,?053,,'
         str(12)='1J0007,1000,3165,4110,5245,6006,7090,8210,9003,'//
     &   ':221,;107,<075,=077,>053,,'
         str(11)='1J0007,1000,3165,4110,5245,6006,7090,8210,9003,'//
     &   ':107,;075,<077,=053,,'
         str(10)='1J0007,1000,3165,4110,5245,6006,7090,8003,9107,'//
     &   ':075,;077,<053,,'
         str(9)='1J0007,1000,3165,4110,5245,6006,7090,8003,9107,'//
     &   ':077,;053,,'
         str(8)='1J0007,1000,3165,4110,5006,6090,7003,8107,9077,'//
     &   ':053,,'
         str(7)='1J0007,1000,3110,4006,5090,6003,7107,8077,9053,,'
         str(6)='1J0007,1000,3110,4006,5090,6003,7107,8077,,'
         str(5)='1J0007,1000,3006,4090,5003,6107,7077,,'
         str(4)='1J0007,1000,3006,4090,5003,6077,,'
         str(3)='1J0007,1000,3006,4003,5077,,'
         str(2)='1J0007,1000,3006,4077,,'
         str(1)='1J0007,1000,3006,,'
         Call rdeblank(str(ncolors),str(ncolors),leng)
         Call esccom(str(ncolors)(1:leng))
      Else
         Print *,' Subroutine INKJET cannot handle ncolors',ncolors
      End If
C
      Return
      End
C+
C ****************************************************************
```

```
C  intcal - Initialize the calc common block.
C-
C ***************************************************************
      Subroutine intcal
      Common /calc/ncont,cmin,cdel
C
      ncont=0
      cmin=0.0
      cdel=0.0
C
      Return
      End
C+
C ***************************************************************
C  intfil - Initializes the parameters used in pattern filling
C           a polygon.
C-
C ***************************************************************
      Subroutine intfil
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1
C
      open='P'
      solid='0'
      filtyp='1'
C
      Return
      End
C+
C ***************************************************************
C  intflg - Initializes flags used in program POLYGON.
C-
C ***************************************************************
      Subroutine intflg
      Common /flags/mcflag
      Common /model/mdflag
      Character mcflag*2
C
      mcflag='M'
      mdflag=0
C
      Return
      End
C+
C ***************************************************************
C  intlab - Initializes the labels associated with the value
C           arrays.
C-
C ***************************************************************
      Subroutine intlab
      Common /labels/label
      Character label(10)*15
C
      nlab=10
      Do 10 i=1,nlab
      label(i)='UNASSIGNED'
10    Continue
C
      Return
      End
C+
C **************************************************************
C  intloc - INITialize LOcation - Initializes the location
C           and boundary arrays.
```

```
C ************************************************************
      Subroutine intloc
      Common /commands/nmax,epsiln,delin,delout
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
C
      Do 10 npoly=1,nmax
      Call delloc(npoly,itest)
   10 Continue
C
      ngbloc=0
      Do 20 j=1,nmax
      jnkloc(j)=0
   20 Continue
C
      Return
      End
C+
C ************************************************************
C  intnew - Initializes the working topology array in the newtopo
C           common block.
C-
C ************************************************************
      Subroutine intnew
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /commands/nmax,epsiln,delin,delout
C
      Do 10 i=1,nmax
      infnew(i)=0
      iupnew(i)=0
      idwnew(i)=0
      ilfnew(i)=0
      irtnew(i)=0
   10 Continue
C
      Return
      End
C+
C ************************************************************
C  intorg - Initializes the variables stored in the
C           common block.
C-
C ************************************************************
      Subroutine intorg
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
C
      iwcorg=0
      jwcorg=0
      nxorgp=0
      nyorgp=0
C
      Return
      End
C+
C ************************************************************
C  intpar - Initializes the parameters in the parm list.
C-
C ************************************************************
      Subroutine intpar
      Common /Parameter/parm(100,10)
      Common /commands/nmax,epsiln,delin,delout
C
      Do 10 i=1,nmax
      Do 10 j=1,10
      param(i,j)=0.0
   10 Continue
C
      Return
      End
C+
C *************************************************************
C  intply - Initializes the POLYLOC common block, used by most of
C           the editing and special function routines.
C-
C *************************************************************
      Subroutine intply
      Common /max/nptmax
      Common /polyloc/nptloc,xloc(100),yloc(100)
C
      nptloc=0
      Do 10 i=1,nptmax
      xloc(i)=0.0
      yloc(i)=0.0
   10 Continue
C
      Return
      End
C+
C *************************************************************
C  intscr - Initializes the parameters used in drawing the grid
C           and polygons on the screen.
C-
C *************************************************************
      Subroutine intscr
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /subscreen/xscrn(2),yscrn(2),xgrd(2),ygrd(2)
      Common /commands/nmax,epsiln,delin,delout
      Character id*56,pgm*8,zmflag*1
C
      xinit=100.0
      yinit=100.0
C
      xsc=0.0
      ysc=0.0
      xstart=1
      ystart=1
C
      icmin=1
      icmax=nc
      irmin=1
      irmax=nr
C
      ncmin=1
      ncmax=nc
      nrmin=1
      nrmax=nr
C
      xscrn(1)=0.0
      yscrn(1)=0.0
      xscrn(2)=4000.0
      yscrn(2)=3000.0
C
      xgrd(1)=0.0
      ygrd(1)=0.0
      xgrd(2)=0.0
      ygrd(2)=0.0
C
      epsiln=1.0e-10
      delin=25.0
      delout=40.0
C
      Return
      End
C+
C ****************************************************************
C  inttmp - Initializes the TEMP array, used by most of the editing
C           and special function routines.
C-
C ****************************************************************
      Subroutine inttmp
      Common /max/nptmax
      Common /temp/ntemp,xtemp(100),ytemp(100)
C
      ntemp=0
      Do 10 i=1,nptmax
      xtemp(i)=0.0
      ytemp(i)=0.0
   10 Continue
C
      Return
      End
C+
C ***************************************************************
C  inttpl - INITial ToPoLogy - Initializes the topology arrays
C           and the garbage collector jnktop.
C-
C ***************************************************************
      Subroutine inttpl
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /jurk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /commands/nmax,epsiln,delin,delout
C
      ngbtop=0
      Do 10 npoly=1,nmax
C
      info(npoly)=0
      iupper(npoly)=0
      idown(npoly)=0
      ileft(npoly)=0
      iright(npoly)=0
C
      jnktop(npoly)=0
   10 Continue
C
      Return
      End
C+
C ***************************************************************
C  intwrk - Initializes work grid (wrkgrd) to dval.
C-
C ***************************************************************
      Subroutine intwrk(dval)
      Common /work/wrkgrd(250000)
C
      Do 10 i=1,250000
      wrkgrd(i)=dval
   10 Continue
C
```

```
      Return
      End
C+
C ***************************************************************
C  intzom - Initializes the zoom stacks ncminz,ncmaxz,nrminz,
C           nrmaxz for the zoom command.
C
C           izoom  = Flag for zoom state of program
C                  = 1 = Yes, program is in zoomed state
C                  = 0 = No, program is not in zoomed state
C
C           nzoom  = Pointer to top of zoom stack, range is from 0-5
C
C           izval  = Pointer used by subroutine unzcom (UNZOOM
C                    COMMAND) to identify zoom values to use in
C                    recall mode.
C-
C ***************************************************************
      Subroutine intzom
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
C
      nzoom=0
      Do 20 i=1,5
      ncminz(i)=0
      ncmaxz(i)=0
      nrminz(i)=0
      nrmaxz(i)=0
   20 Continue
C
      Return
      End
C+
C ****************************************************************
C  invers - Transforms a coordinate (xold,yold) from the zoomed
C           screen to unzoomed screen coordinates (xnew,ynew).
C-
C ****************************************************************
      Subroutine invers(xnew,ynew,xold,yold)
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
C
      xorg=iwcorg+(ncmin-1)*nxorgp
      yorg=jwcorg+(nrmin-1)*nyorgp
C
      xnew=xorg+(xold-iwc0)*nxorgp/nxpix
      ynew=yorg+(yold-jwc0)*nyorgp/nypix
C
      Return
      End
C+
C     IRQUEST   - asks question with real answer
C
C      ireturn = IRQUEST (quest,rval,form,mode)
C
C       ireturn =  -1 if '//' was given as response (user wants out)
C                   0 if no response (user took default)
C                   1 if user responded (returns response in aval)
C
C          quest =  Character string containing question to be asked
C                  (with no ? at the end, it is added by function)
C
C           rval =  Real variable to receive answer
```

```
C                 (used to pass default if one is available)
C
C          form =  Character string containing fortran format to be
C                  used to format the default contained in rval
C
C          mode =  Integer control parameter:
C
C             mode = 0, required response (no default allowed)
C             mode <>0, default allowed
C-
C  ************************************************
      Integer Function irquest(quest,rval,form,mode)
      Character quest*(*),form*(*),rstr2*30
      Character*100 str,form2*10,ans*30,astr*30
      irquest=0
      iqlen=itlen(quest)
      Write (rstr2,form)rval
      irlen=ideblank(rstr2)
      If (mode.NE.0) Then
         str=quest(1:iqlen)//' ['//rstr2(1:irlen)//']?'
         islen=iqlen+irlen+4
         Else
         str=quest(1:iqlen)//'?'
         islen=iqlen+1
         End If
      Write (form2,105)islen
  105 Format ('(x,a',i3,',$)')
   13 Write (6,form2)str
      Read (5,110)ans
  110 Format (a30)
      ialen=ideblank(ans)
      If (ialen.NE.0) Then
         If (ans.EQ.'//') Then
            irquest=-1
            Else
            irquest=1
            Read (ans(1:ialen),120,err=25)rval
  120       Format (f20.0)
            End If
         Else
         If (mode.EQ.0)Go To 13
         End If
      Return
C     couldn't decode
   25 Continue
      Write (6,130)
  130 Format (' Please answer again, I''m expecting a number.')
      Go To 13
      End
C+
C     ITLEN    - Gives length of string without trailing blanks
C
C     ireturn = ITLEN (string)
C
C        ireturn = the length of the string without trailing blanks
C         string = string to have trailing blanks removed from
C
C-
C  ************************************************
      Integer Function itlen(string)
      Character string*(*)
      i=len(string)
   10 Continue
      If (string(i:i).NE.' ')Go To 15
      i=i-1
      If (i.GT.0)Go To 10
   15 Continue
      itlen=i
      Return
      End
C+
C ***************************************************************
C   lincur - Returns the world coordinates of a line (xline,yline)
C            entered using the Envision cross-hair cursor keys.
C
C            The parametors are explained in LINCUR.INF.
C-
C ***************************************************************
      Subroutine lincur(xline,yline,nline,x1,y1,x2,y2,iset,nbrpts,
     &nptmax,itest)
      Dimension xline(nptmax),yline(nptmax)
      Character ans*1
C
      Call curon
      i=0
      itest=0
      nline=0
      ntotal=nbrpts
      If (itest.EQ.0) Then
   10    Continue
         Call gtpnt(ans,x,y)
         If (ans.EQ.'e'.OR.ans.EQ.'f') Then
            If (ntotal.EQ.(nptmax-1).AND.ans.EQ.'e') Then
               Print *,' Last point, only (f/q) are allowed answers'
            Else
               i=i+1
               ntotal=ntotal+1
               xline(i)=x
               yline(i)=y
               Call drwpnt(x,y)
               If (i.EQ.1.AND.(iset.EQ.0.OR.iset.EQ.1))Call
     &         drwlin(x1,y1,x,y)
               If (i.GT.1)Call drwlin(xline(i-1),yline(i-1),x,y)
               If (ans.EQ.'f'.OR.ntotal.EQ.nptmax) Then
                  nline=i
                  itest=1
                  If (iset.EQ.0.OR.iset.EQ.2)Call drwlin(x,y,x2,y2)
               End If
            End If
C
         Else If (ans.EQ.'h') Then
            Call hlpent
C
         Else If (ans.EQ.'q') Then
            nline=i
            itest=-1
C
         Else
            Call errmsg
C
         End If
         If (ntotal.EQ.(nptmax-1).AND.itest.EQ.0) Then
            Print *,i,' coordinates entered, only one more allowed'
         End If
         If (i.GT.nptmax)itest=-1
         If (itest.EQ.0)Go To 10
      End If
C
```

```
      Call curoff
      Return
      End
C+
C ******************************************************************
C  linmou - Returns the world coordinates of a line (xline,yline)
C          entered using the Envision mouse.
C
C          The parameters are explained in LINMOU.INF.
C-
C ******************************************************************
      Subroutine linmou(xline,yline,nline,x1,y1,x2,y2,iset,nbrpts,
     &nptmax,itest)
      Dimension xline(nptmax),yline(nptmax)
      Character ans*1
C
      Call curon
      Call softky('1')
      Call setmou
      Call loadmou
C
      itest=0
      i=0
      nline=0
      ntotal=nbrpts
      If (itest.EQ.0) Then
   10    Continue
         Call getmou(mode,ix,iy)
         x=float(ix)
         y=float(iy)
         If (mode.EQ.1.OR.mode.EQ.2) Then
            If (mode.EQ.1.AND.ntotal.EQ.(nptmax-1)) Then
               Print *,' Last point, only button 2 or buttons 2&3 are
     & allowed'
            Else
               i=i+1
               ntotal=ntotal+1
               xline(i)=x
               yline(i)=y
               Call drwpnt(x,y)
               If (i.EQ.1.AND.(iset.EQ.0.OR.iset.EQ.1))Call
     &         drwlin(x1,y1,x,y)
               If (i.GT.1)Call drwlin(xline(i-1),yline(i-1),x,y)
               If (mode.EQ.2.OR.ntotal.EQ.nptmax) Then
                  nline=1
                  itest=1
                  If (iset.EQ.0.OR.iset.EQ.2)Call drwlin(x,y,x2,y2)
               End If
            End If
C
         Else If (mode.EQ.3) Then
            Call hlpmou
C
         Else If (mode.EQ.23) Then
            nline=1
            itest=-1
C
         Else
            Call errmsg
C
         End If
         If (ntotal.EQ.(nptmax-1).AND.itest.EQ.0) Then
            Print *,i,' coordinates entered, only one more allowed'
         End If
```

```
           If (i.GT.nptmax)itest=-1
           If (itest.EQ.0)Go To 10
        End If
C
        Call softky('0')
        Call curoff
        Return
        End
C+
C ***************************************************************
C  loadmou - Loads the mouse soft key button definitions within
C            the terminals key translation table.
C-
C ***************************************************************
        Subroutine loadmou
        Character esc*1
C
        esc=char(27)
        Call esccom('[A'//esc//'Ky0107\1\2\3\4\5\6\7')
C
        Return
        End
C+
C ***************************************************************
C  loccur - LOCate CURsor - Return world coordinates (ix0,iy0)
C           of Envision cross-hair cursor.
C-
C ***************************************************************
        Subroutine loccur(ix0,iy0)
C
        Call esccom('YC')
        Read (5,20)ix0,iy0
    20  Format (2z4)
C
        Return
        End
C+
C ***************************************************************
C  magmod - Writes out a MAGPOLY model using the POLYGON model
C           information and user entered responses.
C-
C ***************************************************************
        Subroutine magmod(itest)
        Dimension xpoly(100),ypoly(100)
        Common /topology/info(100),iupper(100),idown(100),ileft(100),
       &iright(100)
        Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
        Common /parameter /parm(100,10)
        Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
        Common /original/iwcorg,jwcorg,nxorgp,nyorgp
        Character idplot*40,ifile*50,ifile2*50,quest*80,ifmt*80,
       &id*56,pgm*8,comfil*80,ans*2,ans2*2,name*8,unix*4,uniz*4
        namelist/parms/iplotr,ibody,ifile,ifile2,isqs,datum,xscale,
       &dc,unix,uniz,name,height,ifmt,idplot,iobs,icalc,ires,naxcol
C
        iplotr=9
        naxcol=130
        ibody=0
        iobs=0
        icalc=0
        ires=0
        idplot=' '
        ifile=' '
        ifile2=' '
```

```
      isqs=0
      datum=0.
      xscale=0.
      dc=0.
      unix='kilm'
      uniz='feet'
      name='gridded'
      height=0.
C
C - Ask for name of file containing fieldpoint information.
C
      itest=1
      quest=' Name of file containing fieldpoint information'
      ival=iaquest(quest,ifile,'(a50)',0)
      If (ival.EQ.-1)Go To 100
C
C - Name (type of data read in from ifile).
C
      itest=0
      Do 40 while(itest.EQ.0)
         quest=' Type of data in fieldpoint file (enter H for help)'
         ival=iaquest(quest,name,'(a8)',-8)
C
         If (name(1:1).EQ.'G') Then
            name='gridded'
            height=0.0
            Write (6,45)
   45 Format (2x,'Height of fieldpont grid above the same datum that')
            quest=' is used to reference the body heights'
            ival=irquest(quest,height,'(e16.8)',16)
            itest=1
            If (ival.EQ.-1)itest=-1
         Else If (name.EQ.'H') Then
            Call hlpg2
         Else If (ival.EQ.-1) Then
            itest=-1
         Else
C
C - Ask for format type of data.
C
            itest2=0
            Do 25 while(itest2.EQ.0)
               quest=' Enter format to use when reading in your file'
               ival=iaquest(quest,ifmt,'(a80)',0)
               If (ival.EQ.-1) Then
                  itest=-1
                  itest2=-1
               Else If (ifmt.NE.' ') Then
                  itest=1
                  itest2=1
               Else
                  Call errmsg
               End If
   25       End Do
         End If
   40 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for the name of standard grid containing the heights of
C   fieldpoints.
C
      quest=' Name of standard grid containing heights of fieldpoints'
      ival=iaquest(quest,ifile2,'(a50)',0)
      If (ival.EQ.-1)Go To 100
```

```
C
C - Ask for print out identifier.
C
      quest=' Identifier for printer output'
      ival=iaquest(quest,idplot,'(a40)',0)
      If (ival.EQ.-1)Go To 100
C
C - Ask if constant should be added to calculated anomalies.
C
      itest=0
      Do 20 while(itest.EQ.0)
         quest=' Add a constant to the calculated anomalies (y/n/q)'
         ival=iaquest(quest,ans,'(a2)',0)
C
         If (ans.EQ.'Y') Then
            quest=' Constant to be added'
            ival=irquest(quest,datum,'(e16.8)',0)
            itest=1
            If (ival.EQ.-1)itest=-1
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else If (ans.EQ.'N') Then
            itest=1
            datum=0.0
         Else
            Call errmsg
         End If
   20 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask lsqs determination.
C
      itest=0
      Do 30 while(itest.EQ.0)
         quest=' Least-squares comparison/read observed values
     & (y/n/h/q)'
         ival=iaquest(quest,ans2,'(a2)',0)
C
         If (ans2.EQ.'Y') Then
            itest2=0
            Do 35 while(itest2.EQ.0)
               lsqs=0
               quest=' What value for LSQS (0-7 integer, // to quit)'
               ival=iiquest(quest,lsqs,'(i2)',-2)
C
               If (lsqs.GE.0.AND.lsqs.LE.7) Then
                  itest2=1
                  itest=1
               Else If (ival.EQ.-1) Then
                  itest2=-1
                  itest=-1
               Else
                  Call errmsg
               End If
   35       End Do
         Else If (ans2.EQ.'N') Then
            lsqs=0
            itest=1
         Else If (ans2.EQ.'H') Then
            Call hlpmgl
         Else If (ans2.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   30 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for units in x&y directions.
C
      itest=0
      Do 50 while(itest.EQ.0)
         unix='KILM'
         quest=' Units in x&y direction (FEET/KILF/MILE/METR/KILM/H/Q)'
         ival=iaquest(quest,unix,'(a4)',-4)
C
         itest=1
         If (unix.EQ.'FEET') Then
            unix='feet'
         Else If (unix.EQ.'MILE') Then
            unix='mile'
         Else If (unix.EQ.'KILF') Then
            unix='kilf'
         Else If (unix.EQ.'METR') Then
            unix='metr'
         Else If (unix.EQ.'KILM') Then
            unix='kilm'
         Else If (unix.EQ.'H') Then
            Call hlpg3
            itest=0
         Else If (unix.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
            itest=0
         End If
   50 End Do
      If (itest.EQ.-1)Go To 100
C
C - Ask for units in z direction (height).
C
      itest=0
      Do 60 while(itest.EQ.0)
         uniz='FEET'
         quest=' Units in z direction (FEET/KILF/MILE/METR/KILM/H/Q)'
         ival=iaquest(quest,uniz,'(a4)',-4)
C
         itest=1
         If (uniz.EQ.'FEET') Then
            uniz='feet'
         Else If (uniz.EQ.'MILE') Then
            uniz='mile'
         Else If (uniz.EQ.'KILF') Then
            uniz='kilf'
         Else If (uniz.EQ.'METR') Then
            uniz='metr'
         Else If (uniz.EQ.'KILM') Then
            uniz='kilm'
         Else If (uniz.EQ.'H') Then
            Call hlpg3
            itest=0
         Else If (uniz.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
            itest=0
         End If
   60 End Do
```

```
      If (itest.EQ.-1)Go To 100
C
C - Ask for print switch information.
C
      itest=0
      Do 65 while(itest.EQ.0)
         iflag=0
         quest=' Print switch for body information (1/0)'
         ival=iiquest(quest,iflag,'(i1)',1)
C
         If (ival.EQ.-1) Then
            itest=-1
         Else If (iflag.EQ.0.OR.iflag.EQ.1) Then
            itest=1
         Else
            Call errmsg
         End If
   65 End Do
      If (itest.LT.0)Go To 100
C
C - Ask which parm from the parm arrays to use when creating
C   this model.
C
      Call askmvl(ione,itwo,ithree,ifour,ifive,isix,itest)
      If (itest.EQ.-1)Go To 100
C
C - Ask for the name to call this MAGPOLY file (comfil)
C
   70 Continue
      quest=' Name to call command file to run MAGPOLY'
      ival=iaquest(quest,comfil,'(a80)',0)
      If (ival.EQ.-1)Go To 100
      If (comfil.EQ.' ')Go To 70
C
C - Open the model file and write out the model.
C
      Open (11,file=comfil,status='new',carriagecontrol='list')
      Write (11,nml=parms)
      Call fndtop(ntop)
      next=ntop
      itest=0
      delxcn=dx/nxorgp
      delycn=dy/nyorgp
      Do 150 while(itest.EQ.0)
         ngon=next
         npoly=info(ngon)
         nbrpts=numply(npoly)
         If (nbrpts.GT.0) Then
            Do 120 i=1,nbrpts
            xpoly(i)=xo+(xscr(npoly,i)-iwcorg)*delxcn
            ypoly(i)=yo+(yscr(npoly,i)-jwcorg)*delycn
  120       Continue
C
C - Find the handedness of the polygon
C       gamma > 0.0  = polygon is counter-clockwise
C       gamma < 0.0  = polygon is clockwise
C
            Call totalt(gamma,xpoly,ypoly,nbrpts)
C
            If (gamma.GT.0) Then
               nstart=nbrpts
               nend=1
               istep=-1
            Else
```

```
               nstart=1
               nend=nbrpts
               istep=1
            End If
C
C - If the body has a parent, need to determine if the parent-body/
C   current body overlap.
C
            iup=iupper(ngon)
            If (iup.GT.0) Then
               nplyup=info(iup)
C
               partop=parm(nplyup,ione)
               parbot=parm(nplyup,itwo)
               bodtop=parm(npoly,ione)
               bodbot=parm(npoly,itwo)
C
C - We need only remove a slab if the parent body and current body
C   overlap in the z direction (height).
C
               If (((bodtop.LE.partop).AND.(bodbot.GE.parbot)).OR.
     &         ((bodtop.GT.partop).AND.(bodbot.LT.partop)).OR.
     &         ((bodtop.GT.parbot).AND.(bodbot.LT.parbot))) Then
C
                  If (bodtop.LE.partop) Then
                     ztop=amin1(bodtop,partop)
                  Else
                     ztop=partop
                  End If
C
                  If (bodbot.GE.parbot) Then
                     zbot=amax1(bodbot,parbot)
                  Else
                     zbot=parbot
                  End If
C
C - Write out the slab to be removed.
C
                  Write (11,*)nbrpts,iflag,-parm(nplyup,ithree),
     &parm(nplyup,ifour),parm(nplyup,ifive),parm(nplyup,isix),ztop,zbot
                  Write (11,*)(xpoly(i),ypoly(i),i=nstart,nend,istep)
               End If
            End If
            Write (11,*) nbrpts,iflag,parm(npoly,ithree),
     & parm(npoly,ifour),parm(npoly,ifive),parm(npoly,isix),
     & parm(npoly,ione),parm(npoly,itwo)
            Write (11,*)(xpoly(ii),ypoly(ii),ii=nstart,nend,istep)
         End If
C
C - Get next polygon
C
         Call walk(next,ngon,itest2)
         If (next.EQ.0.OR.itest2.LE.0)itest=1
  150 End Do
      Close (11)
C
  100 Continue
      Return
      End
C+
C ****************************************************************
C  modin - MODel INput - Asks for the name of the model, clears
C          the model arrays and reads in the model.
C-
```

```
C ****************************************************************
      Subroutine modin(ilast)
      Dimension xpoly(100),ypoly(100)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /names/grdnam,modnam,modgrd
      Common /misc/ncol,nrow,first,ntop
      Common /commands/nmax,epsiln,delin,delout
      Common /model/mdflag
      Character first*1,modgrd*80,grdnam*80,modnam*80
C
   10 Continue
C
C - Ask for the name of the model
C
      Call askmdl(modnam,itest)
      If (itest.EQ.-1.OR.itest.EQ.0)Go To 100
C
C - Clear model arrays for reading
C
      Call intloc
      Call inttpl
      Call intnew
C
C - Read in the model
C
      Call readmd(itest)
      If (itest.EQ.-1)Go To 10
      Call oldnew
      mdflag=1
C
      Call fndtop(ntop)
      itest=0
      i=1
      ncount=0
      Do 50 while(itest.EQ.0.AND.i.GE.1.AND.i.LE.nmax)
         npoly=info(i)
         If (npoly.GT.0) Then
            nbrpts=numply(npoly)
            Do 40 j=1,nbrpts
            xpoly(j)=xscr(npoly,j)
            ypoly(j)=yscr(npoly,j)
   40       Continue
            Call fndbbx(xmin,xmax,ymin,ymax,xpoly,ypoly,nbrpts,delout)
            xminbx(npoly)=xmin
            xmaxbx(npoly)=xmax
            yminbx(npoly)=ymin
            ymaxbx(npoly)=ymax
            Call fndbpl(npoly,itest)
            itest=0
            ncount=ncount+1
            If (ncount.EQ.ntotal)itest=1
         End If
         i=i+1
   50 End Do
C
  100 Continue
      Return
      End
C+
C ****************************************************************
C  modout - MODel OUTput - Asks for the name of the model and
```

```
C           writes it out.
C-
C ******************************************************************
      Subroutine modout(itest)
      Common /names/grdnam,modnam,modgrd
      Character grdnam*80,modnam*80,modgrd*80
C
C - Ask for the name of the model
C
      Call askmdl(modnam,itest)
      If (itest.EQ.-1.OR.itest.EQ.0)Go To 100
C
C - Write out the model
C
      Call wrtmod(itest)
  100 Continue
      Return
      End
C+
C ******************************************************************
C  modpnt - Determines the corner number of a polygon picked.
C-
C ******************************************************************
      Subroutine modpnt(ncorn,dist,x,y,ans,npoly,mcur)
      Dimension xply(100),yply(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /commands/nmax,epsiln,delin,delout
      Character mcur*2,ans*2
C
      nbrpts=numply(npoly)
      If (nbrpts.GT.0) Then
         ans='N'
         Do 40 while(ans.EQ.'N')
            Call hlpcpl
            Call retpnt(x,y,mcur,ierr2)
            If (ierr2.GE.1) Then
               If (izoom.EQ.1)Call invers(x,y,x,y)
               Do 35 k=1,nbrpts
               xply(k)=xscr(npoly,k)
               yply(k)=yscr(npoly,k)
   35          Continue
               icon=1
               Call clspnt(ncorn,dist,x,y,xply,yply,nbrpts,delout,icon)
               If (ncorn.EQ.0) Then
                  Call wrtmsg(' Could not find corner near your
     & point...try again')
               Else
                  ans='Y'
               End If
            Else If (ierr2.EQ.-1) Then
               ans='Q'
            End If
   40    End Do
      Else
         Call wrtmsg(' Polygon number passed to MODPNT has no corners')
         ans='Q'
      End If
C
      Return
      End
C+
C ******************************************************************
```

```
C  mouexm - MOUse EXecution Mode - Selects Execution mode for
C           Envision mouse, (0=Point mode,1=Multiple mode).
C-
C **************************************************************
      Subroutine mouexm(mode)
      Character mode*1,com*3
C
      com='IZ'//mode
      Call esccom(com)
C
      Return
      End
C+
C **************************************************************
C  moulin - MOUse LINear - Places Envision mouse in linear res-
C           ponse mode.
C-
C **************************************************************
      Subroutine moulin
C
      Call esccom('IV')
C
      Return
      End
C+
C **************************************************************
C  mousop - MOUse Suspend OPerations - Suspends mouse operation
C           until linear or logarithmic response mode is re-
C           selected.
C-
C **************************************************************
      Subroutine mousop
C
      Call esccom('IX')
C
      Return
      End
C+
C ***************************************************************
C  movcur - Moves the graphics cross-hair cursor to world coordinates
C           (ix,iy).
C-
C ****************************************************************
      Subroutine movcur(ix,iy)
      Character wcbp*5,com*6
C
      com='P'//wcbp(ix,iy)
      Call esccom(com)
C
      Return
      End
C+
C **************************************************************
C  movply - Allows the moving of a polygon from one location to
C           another part of the screen.
C-
C **************************************************************
      Subroutine movply(itest)
      Dimension parstore(10)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /parameter /parm(100,10)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /flags/mcflag,votflg
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /gridspecs/ id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /scale/ xsc,ysc,xstart,ystart,xinit,yinit
      Common /original/ iwcorg,jwcorg,nxorgp,nyorgp
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /state/lflast,lftin,lftout,ndpstn,iup
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character id*56,pgm*8,votflg*2,mcflag*2,mcur*2,ans*2,ans2*2
C
      Call enhmsg('*** Move polygon mode ***')
C
      itest=0
C
C - Test the number of polygons ntotal, exit if <=0.
C
      If (ntotal.GT.0) Then
C
C - If cursor type has not been selected prompt for type.
C
         If (mcflag.EQ.'N') Then
            Call askmoc(mcur)
            If (mcur.EQ.'Q') Then
               itest=-1
            Else
               mcflag=mcur
            End If
         Else
            mcur=mcflag
         End If
C
C - Print help message
C
         If (votflg.EQ.'V')Call hlpmov
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
         Do 10 while(itest.EQ.0)
         iflag=0
         If (mcur.EQ.'M'.OR.mcur.EQ.'C') Then
C
C - Initial the temp arrary.
C
            call inttmp
C
C - Let user pick polygon and return npoly,ncorn,x,y.
C
            Call pckply(npoly,ncorn,x,y,ans,mcur,ierr)
C
C - Ask if corner point of polygon picked should be used.
C
            If (ans.EQ.'Y') Then
               nbrpts=numply(npoly)
               Call pckpnt(ncorn,dist,x,y,ans2,npoly,mcur)
               If (ans2.EQ.'Q') Then
                  itest=-1
                  Go To 20
               End If
C
C - Message about repositioning cursor to new location
C
               Call hlpmv2
               Call retpnt(xscnew,yscnew,mcur,ierr)
               If (ierr.EQ.-1) Then
                  itest=-1
                  Go To 20
               End If
C
               If (izoom.EQ.1) Then
                  Call invers(xnew,ynew,xscnew,yscnew)
               Else
                  xnew=xscnew
                  ynew=yscnew
               End If
C
C - Find the amount that polygon should be moved by (delx,dely).
C
               delx=xnew-x
               dely=ynew-y
C
C - Store new polygon
C
               ntemp=nbrpts
               Do 30 i=1,ntemp
               xtemp(i)=xscr(npoly,i)+delx
               ytemp(i)=yscr(npoly,i)+dely
   30          Continue
C
C - Test if polygon will be off of unzoomed grid.
C
               xlfunz=xinit
               xrgunz=xinit+nc*nxorgp
               ybtunz=yinit
               ytpunz=yinit+nr*nyorgp
               Call testoff(xtemp,ytemp,ntemp,xlfunz,xrgunz,ybtunz,
    & ytpunz,intotal)
               If (intotal.LE.0) Then
                  If (intotal.EQ.0) Call wrtmsg(' Error, polygon will
    & off of unzoomed grid')
                  Go To 20
               End If
C
C - Store the parameter informatiion.
C
               Do 35 ii=1,10
               parstore(ii)=parm(npoly,ii)
   35          Continue
C
C - Test the new polygon in the xtemp,ytemp array and fit into
C   the topology structure.
C
               Call testopo(npoly,itest2)
               If (itest2.GE.1) Then
                  npoly2=info(ndpstn)
                  Do 40 iii=1,10
                  parm(npoly2,iii)=parstore(iii)
   40             Continue
                  iflag=1
               End If
               itest=1
```

```
C
   20          Continue
               If (iflag.EQ.0) Then
                  Call setclr(plyclr)
                  Call setfil(open)
                  Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
               End If
            Else If (ans.EQ.'N') Then
               itest=0
            Else
               itest=-1
            End If
            Call setfil(solid)
            If (ierr.EQ.-1)itest=-1
         Else If (mcur.EQ.'Q') Then
            itest=-1
         End If
   10    End Do
      Else
         Call wrtmsg('  Sorry no polygons')
      itest=-1
      End If
C
      Return
      End
C+
C ************************************************************
C  movpnt - Allows the moving of individual points of a polygon.
C-
C ************************************************************
      Subroutine movpnt(itest)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /parameter /parm(120,10)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /flags/mcflaq,votflg
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character votflg*2,mcflag*2,mcur*2,ans*2,ans2*2
C
      Call enhmsg('*** Move point mode ***')
C
      itest=0
C
C - Test the number of polygons ntotal, exit if <=0.
C
      If (ntotal.GT.0) Then
C
C - If cursor type has not been selected prompt for type.
C
         If (mcflag.EQ.'N') Then
            Call askmoc(mcur)
            If (mcur.EQ.'Q') Then
               itest=-1
            Else
```

```
          mcflag=mcur
        End If
      Else
        mcur=mcflag
      End If
C
C - Print help message if user requested verbose answers.
C
      If (votflg.EQ.'V') Call hlpmvp
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
      Do 10 while(itest.EQ.0)
      iflag=0
      iflag2=0
      If (mcur.EQ.'M'.OR.mcur.EQ.'C') Then
C
C - Initialize the temp arrays.
C
        Call inttmp
C
C - Find polygon, enhance and return npoly,ncorn,x,y.
C
        Call pckply(npoly,ncorn,x,y,ans,mcur,ierr)
C
C - Next ask if corner picked in pckply (ncorn = x,y) should be
C   used.
C
        If (ans.EQ.'Y') Then
          nbrpts=numply(npoly)
          Call pckpnt(ncorn,dist,x,y,ans2,npoly,mcur)
          If (ans2.EQ.'Q') Then
            itest=-1
            Go To 20
          End If
C
C - Now undraw the connecting lines to ncorn.
C
          If (ncorn.EQ.1) Then
            x1=xscr(npoly,nbrpts)
            y1=yscr(npoly,nbrpts)
            x3=xscr(npoly,2)
            y3=yscr(npoly,2)
          Else If (ncorn.EQ.nbrpts) Then
            x1=xscr(npoly,nbrpts-1)
            y1=yscr(npoly,nbrpts-1)
            x3=xscr(npoly,1)
            y3=yscr(npoly,1)
          Else
            x1=xscr(npoly,ncorn-1)
            y1=yscr(npoly,ncorn-1)
            x3=xscr(npoly,ncorn+1)
            y3=yscr(npoly,ncorn+1)
          End If
          x2=xscr(npoly,ncorn)
          y2=yscr(npoly,ncorn)
C
          If (izoom.EQ.1) Then
            Call trans(x1,y1,x1,y1)
            Call trans(x2,y2,x2,y2)
            Call trans(x3,y3,x3,y3)
          End If
C
```

```
          Call setclr(black)
          Call drwlin(x1,y1,x2,y2)
          Call drwlin(x2,y2,x3,y3)
C
C - Message about repositioning cursor to new location
C
          Call hlpmv2
          Call retpnt(xscnew,yscnew,mcur,ierr2)
          If (ierr2.LE.-1) Then
            itest=-1
            Go To 20
          End If
c
c - If program is in zoom state unzoom the screen location.
c
          If (izoom.EQ.1) Then
            Call invers(xnew,ynew,xscnew,yscnew)
          Else
            xnew=xscnew
            ynew=yscnew
          End If
C
C - Draw the connecting lines to the new corner
C
          Call setclr(white)
          Call drwlin(x1,y1,xscnew,yscnew)
          Call drwlin(xscnew,yscnew,x3,y3)
C
C - Store new polygon
C
          ntemp=nbrpts
          Do 30 i=1,ntemp
            xtemp(i)=xscr(npoly,i)
            ytemp(i)=yscr(npoly,i)
  30      Continue
          xtemp(ncorn)=xnew
          ytemp(ncorn)=ynew
C
C - Test the new polygon in xtemp,ytemp array to see if it is
C   self-crossing.
C
          Call selftest(iflag3,nside1,nside2,xtemp,ytemp,ntemp)
          If (iflag3.EQ.-1) Then
            Call wrtmsg(' Error, polygon is self-crossing.')
            iflag2=1
            Go To 20
          End If
C
C - Test the new polygon in the xtemp,ytemp array and fit into
C   the topology structure.
C
          Call testopo(npoly,itest2)
          If (itest2.GE.0) Then
            iflag=1
          Else
            iflag2=1
          End If
          itest=1
C
  20      Continue
          If (iflag.EQ.0) Then
            Call setclr(plyclr)
            Call setfil(open)
            Call drwclp(xloc,ylcc,nptloc,xleft,xright,ybot,ytop)
```

```
                  If (iflag2.EQ.1) Then
                     Call setclr(black)
                     Call drwlin(x1,y1,xscnew,yscnew)
                     Call drwlin(xscnew,yscnew,x3,y3)
                     Call setclr(plyclr)
                  End If
               End If
            Else If (ans.EQ.'N') Then
               itest=0
            Else
               itest=-1
            End If
            Call setfil(solid)
            If (ierr.EQ.-1)itest=-1
         Else If (mcur.EQ.'Q') Then
            itest=-1
         End If
   10 End Do
      Else
         call wrtmsg(' Sorry, no polygons')
         itest=-1
      End If
C
      Return
      End
C+
C *************************************************************
C  msgall - Message displayed at user's terminal giving info about
C           assigning values to a polygon.
C-
C *************************************************************
      Subroutine msgall
C
      Write (6,10)
   10 Format (/,' You will now be asked to assign parameters to the
     & polygon drawn in white.',/,'   Enter a carriage return <CR> to
     & use the default answer in brackets [].',/)
C
      Return
      End
C+
C *************************************************************
C  msgclr - Prints message at user's terminal on first time into
C           subroutine CLRSGD (Color Standard Grid).
C-
C *************************************************************
      Subroutine msgclr
C
      Write (6,10)
   10 Format (/,' To draw the grid in color I first need some',
     &' facts about it')
C
      Return
      End
C+
C *************************************************************
C  msgent - Prints message on user's terminal on how to use the
C           a screen coordinate.
C-
C *************************************************************
      Subroutine msgent
C
      Write (6,30)
   30 Format (/,'  To enter a polygon use the cursor keys to position',
     &/,' the cross-hairs.  Then type:',
     &//,'        e = enter the current cross-hair location',
     &/,'        f = finish drawing the polygon (and e)',
     &/,'        h = help, repeats this help message',
     &/,'        q = quit')
C
      Return
      End
C+
C ****************************************************************
C  msggvl - Message explaining information that subroutine askgvl
C           asks user for.
C-
C ****************************************************************
      Subroutine msggvl
C
      Write (6,10)
   10 Format (/,' You will now enter the numbers of the labels',
     &' assigned to the following parameters:',/,
     &/,'       (1) Height of Top of body',
     &/,'       (2) Height of Bottom of body',
     &/,'       (3) Density Contrast of body',/)
C
      Return
      End
C+
C ****************************************************************
C  msglab - Displays on the user's terminal the labels currently
C           assigned to the parameter arrays parm*.
C-
C ****************************************************************
      Subroutine msglab
      Common /labels/label
      Character label(10)*15
C
C - Display the current labels.
C
      Call wrtmsg(' The parameter labels are:')
      Do 8 i=1,10
      Write (6,12)i,label(i)
   12 Format ('    Label ',i2,': ',a15)
    8 Continue
C
      Return
      End
C+
C ***************************************************************
C  msgmo1 - MeSsaGe MOuse 1 - First help message on using the
C           Envision mouse.
C-
C ***************************************************************
      Subroutine msgmo1
      Write (6,10)
   10 Format (//,
     &'  When holding the mouse in the right hand the  mouse'
     &' buttons are numbered',/,' from left to right.  Button 1 is'
     &' actuated by the index finger',/,' and button 3 is actuated by'
     &' the ring finger.',/)
      Return
      End
C+
C ***************************************************************
C  msgmo2 - MeSsaGe MOuse 2 - Second help message on using the
C           Envision mouse.
```

```
C-
C ************************************************************************
      Subroutine msgmo2
C
      Write (6,10)
   10 Format (/,' Use the mouse to position the cross-hair cursor',
     &/,' to the desired location on the screen, then type:')
C
      Return
      End
C+
C ************************************************************************
C  msgmo3 - MeSsaGe MOuse 3 - Third help message on using the
C           Envision mouse.
C-
C ************************************************************************
      Subroutine msgmo3
C
      Write (6,10)
   10 Format (/,' KEY=KEYBOARD, MOUSE=MOUSE BUTTON',
     &/,'      (KEY)   (MOUSE)',
     &/,'        e       1      = Enter cross-hair cursor screen',
     &' location',
     &/,'        f       2      = Enter screen location and finish',
     &' drawing polygon',
     &/,'                         (in ADD polygon mode only)',
     &/,'        h       3      = Help message',
     &/,'        q     2 & 3    = Quit',/)
C
      Return
      End
C+
C ************************************************************************
C  msgmv1 - Message explaining information that subroutine askmv1
C           asks user for.
C-
C ************************************************************************
      Subroutine msgmv1
C
      Write (6,10)
   10 Format (/,' You will now enter the numbers of the labels',
     &' assigned to the following parameters:',/,
     &//,'      (1) Height of Top of body',
     &/,'      (2) Height of Bottom of body',
     &/,'      (3) Volume magnetic susceptibility (emuX100000)',
     &/,'      (4) Remanent or total volume magnetization (emuX100000)'
     &,
     &/,'      (5) Declination of remanent or total magnetization',
     &' in degrees, measured',/,
     &'           positive clockwise from the direction of the y-axis',
     &/,'      (6) Inclination of remanent or total magnetization',
     &' in degrees, measured',/,
     &'           positive downward from the horizontal plane',/)
C
      Return
      End
C+
C ************************************************************************
C  msgsp2 - Prints message at user's terminal about selecting the
C           corner of the polygon to start adding points from.
C-
C ************************************************************************
      Subroutine msgsp2
C
```

```
      Write (6,10)
   10 Format (/,' Now enter one of the endpoints of this side to start'
     &,
     &' drawing from and then',/,' enter the location(s) of the',
     &' point(s) to be added.',/)
C
      Return
      End
C+
C ************************************************************
C  msgspm - MeSsaGe Single Point Mode - Displays enhanced mess-
C           age at user's terminal on entry to single point mode
C           using Envision terminal.
C-
C ************************************************************
      Subroutine msgspm
C
      Call envbcl('1')
      Write (6,10)
   10 Format (/,4x,'*** Enter screen location ***   ')
      Call envbcl('0')
C
      Return
      End
C+
C ************************************************************
C  msgspt - Prints message at user's terminal about selecting
C           the corner of polygon to use.
C-
C ************************************************************
      Subroutine msgspt
C
      Write (6,10)
   10 Format (/,' Position the cross-hair cursor to the side of the',
     &' polygon where point(s)',/,' will be added and enter this',
     &' side using the mouse or keyboard keys.',/)
C
      Return
      End
      subroutine msgstd
c
      write(6,10)
10    format(/,' You have the option to use a prexisting grid',
     & ' when writing out',/,' the standard grid.',/)
c
      return
      end
C+
C *************************************************************
C  msgstk - Message for zoom stack release.
C-
C *************************************************************
      Subroutine msgstk
C
      Write (6,10)
   10 Format (/,'  The zoom stack is full, there are several ways',
     &' to release (free)',/,' a space in the stack.',/)
C
      Return
      End
C+
C *************************************************************
C  msgsub - Prints message at user's terminal about methode for
C           entering the subgrid boundaries in zoom mode.
```

```
C           Called by subroutine asktyp (ASK TYPE).
C-
C ************************************************************
      Subroutine msgsub
C
      Write (6,10)
   10 Format (/,' The subgrid boundaries may be entered',
     &' by one of two methods:',/,
     &/,'     1 = curs r/mouse entered subgrid locations',
     &/,'     2 = prompt for locations of subgrid ',
     &'(ncmin,ncmax,nrmin,nrmax)',
     &/,'     0 or // returns to zoom command level',/)
C
      Return
      End
C+
C ************************************************************
C  msgval - MeSsaGe paramter - Prints a message at the user's term-
C           inal about selecting the paramter to use when reset-
C           ting the grid.
C-
C ************************************************************
      Subroutine msgval
C
      Write (6,10)
   10 Format (/,' For outputting the Denver standard grid, select',
     &' the label number of',/,'    a set of parameters to use when',
     &' resetting the grid values.',/)
C
      Return
      End
      subroutine msgwpg
c
      write (6,10)
10    format(/,' Let''s try again to find your polygon')
c
      return
      end
C+
C ************************************************************
C  ncbp - Number coordinate byte packing.  Converts a single
C         integer number to the Envision terminal code.
C          Author: Robert Simpson.
C-
C ************************************************************
      Character*3 Function ncbp(i)
      Character blank*1
      Parameter (imax=16284)
      Parameter (imin=0)
      Parameter (blank=' ',iblank=ichar(' '))
C
C - Force i into bounds...
      iin=min(imax,max(imin,i))
C - Get hi and lo bytes and offset with blank...
      ilo=mod(iin,64)+iblank
      imed=iin/64+iblank
C - Put bytes together...
      ncbp=blank//char(imed)//char(ilo)
C
      Return
      End
C+
C ************************************************************
C  newold - Copies the topology structure store in the newtopo
```

```
C          common blocks onto the old topology stored in the
C          topology common block.  If a polygon passes a test
C          we do this to store the new structuring of the polygons.
C-
C ***************************************************************
      Subroutine newold
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /commands/nmax,epsiln,delin,delout
C
      Do 10 i=1,nmax
      info(i)=infnew(i)
      iupper(i)=iupnew(i)
      idown(i)=idwnew(i)
      ileft(i)=ilfnew(i)
      iright(i)=irtnew(i)
   10 Continue
C
      Return
      End
C+
C ***************************************************************
C  oldnew - Copies the old topology, stored in the common block
C          topology, onto the new topology stored in the common
C          block newtopo.  This is done to restore the original
C          topology of the polygons, to the working topology
C          arrays (newtopo), if a polygon crosses another during
C          testing.
C-
C ***************************************************************
      Subroutine oldnew
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /commands/nmax,epsiln,delin,delout
C
      Do 10 i=1,nmax
      infnew(i)=info(i)
      iupnew(i)=iupper(i)
      idwnew(i)=idown(i)
      ilfnew(i)=ileft(i)
      irtnew(i)=iright(i)
   10 Continue
C
      Return
      End
C+
C **************************************************************
C  outcom - OUTput COMmand mode - Driver for controlling the
C          output files generated by POLYGON for use by other
C          modeling programs (PFMAG3D, PFGRAV3D, MAGPOLY, GRAV-
C          POLY).  Options are:
C
C              p = Plouff/Godson format for GRAVPOLY/MAGPOLY
C              s = Standard grid for pfmag3d, pfgrav3d, ...
C              h = Help
C              q = Quit and return to Polygon command level
C
C-
C **************************************************************
      Subroutine outcom(itest)
      Character quest*80,ans*2
C
      Call enhmsg('*** Output file mode ***')
C
      itest=0
      ierror=0
      Do 10 while(itest.EQ.0)
         ans='h'
         quest=' Output file mode (p/s/h/q)'
         ival=iaquest(quest,ans,'(a2)',-2)
C
         If (ans.EQ.'P'.OR.ans.EQ.'p') Then
            Call plfout(ierror)
         Else If (ans.EQ.'S'.OR.ans.EQ.'s') Then
            Call stdout(ierror)
         Else If (ans.EQ.'H'.OR.ans.EQ.'h') Then
            Call hlpout
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1.OR.ans.EQ.'q') Then
            itest=1
         Else
            Call errmsg
         End If
         If (ierror.EQ.-1) Then
            Call wrtmsg('  Error encountered')
            ierror=0
            itest=-1
         End If
   10 End Do
C
      Return
      End
C+
C ************************************************************o
C  pckply - Allows the user to position the cross-hair cursor to
C           locate a polygon, enhances the polygon if found, and
C           returns the polygon & corner number, and x,y screen
C           coordinates (unzoomed).
C-
C o************************************************************
      Subroutine pckply(npoly,ncorn,x,y,ans,mcur,itest)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /flags/mcflag,votflg
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character votflg*2,mcflag*2,mcur*2,ans*2
C
      itest=0
      npoly=0
      ncorn=0
C
C - If cursor type has not been selected prompt for type.
C
      If (mcflag.EQ.'N') Then
         Call askmoc(mcur)
         If (mcur.EQ.'Q') Then
            itest=-1
         Else
```

```
            mcflag=mcur
         End If
      Else
         mcur=mcflag
      End If
C
      Call fndtop(ntop)
      If (ntop.GE.1) Then
         lpoly=info(ntop)
      Else
         itest=-1
      End If
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
      Do 10 while(itest.EQ.0)
         Call intply
         Call retpnt(x,y,mcur,ierr)
         If (ierr.GE.1) Then
            If (izoom.EQ.1)Call invers(x,y,x,y)
            Call tester(npoly,ncorn,lpoly,x,y,ierr)
            If (npoly.LE.0) Then
               Call wrtmsg('  Could not find your polygon...try again.')
               ierr=0
            End If
C
            If (ierr.GT.0) Then
C
C - Enhance polygon and prompt.
C
               Call setclr(white)
               Call setfil(open)
               nptloc=numply(npoly)
               If (izoom.EQ.1) Then
                  Do 20 i=1,nptloc
                  Call trans(xloc(i),yloc(i),xscr(npoly,i),
     & yscr(npoly,i))
  20              Continue
               Else
                  Do 25 j=1,nptloc
                  xloc(j)=xscr(npoly,j)
                  yloc(j)=yscr(npoly,j)
  25              Continue
               End If
               Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
C
C - Ask if enhanced polygon is correct one.
C
               Call askenh(ans)
               If (ans.EQ.'Y') Then
                  itest=1
               Else
                  Call setclr(plyclr)
                  Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
                  If (ans.EQ.'Q') Then
                     npoly=0
                     ncorn=0
                     x=0.0
                     y=0.0
                     itest=-1
                  End If
               End If
               Call setfil(solid)
            Else If (ierr.LT.0) Then
               Itest=-1
            End If
         Else
            Itest=-1
         End If
   10 End Do
      If (itest.EQ.-1)ans-'Q'
C
      Return
      End
C+
C ************************************************************
C  pckpnt - Determines the corner number of a polygon picked.
C-
C ************************************************************
      Subroutine pckpnt(ncorn,dist,x,y,ans,npoly,mcur)
      Dimension xply(100),yply(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /commands/nmax,epsiln,delin,delout
      Character mcur*2,ans*2
C
      nbrpts=numply(npoly)
      If (nbrpts.GE.1) Then
         Call askpnt(ans)
         Do 40 while(ans.EQ.'N')
            Call hlpcpl
            Call retpnt(x,y,mcur,ierr2)
            If (ierr2.GE.1) Then
               If (izoom.EQ.1)Call invers(x,y,x,y)
               Do 35 k=1,nbrpts
               xply(k)=xscr(npoly,k)
               yply(k)=yscr(npoly,k)
   35          Continue
               icon=1
               Call clspnt(ncorn,dist,x,y,xply,yply,nbrpts,
     & delout,icon)
               If (ncorn.EQ.0) Then
                  Call wrtmsg(' Could not find corner near you point...
     &try again')
               Else
                  ans='Y'
               End If
            Else If (ierr2.EQ.-1) Then
               ans='Q'
            End If
   40    End Do
      Else
      Call wrtmsg(' Polygon number passed to PCKPNT has no corners')
         ans='Q'
      End If
C
      Return
      End
C+
C *************************************************************
C  pdcom  - Driver for drawing grid on terminal and polygon draw-
C           ing mode (PLYCOM).
C-
C *************************************************************
      Subroutine pdcom(itest)
      Common /misc/ncol,nrow,first,ntop,ifirst
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /model/mdflag
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character plyclr*1,black*1,white*1,open*1,solid*1,filtyp*1,first*1
C
      itest=1
      itest2=1
      If (ifirst.EQ.1) Then
         Call intscr
         Call clrsgd(1,itest2)
         If (itest2.GE.0) Then
            Call clrply
            iwcorg=iwc0
            jwcorg=jwc0
            nxorgp=nxpix
            nyorgp=nypix
            Call setbnd
            If (mdflag.EQ.1) Then
               Call setclr(plyclr)
               Call setfil(open)
               Call drawwalk
               Call setfil(solid)
            End If
         End If
      End If
      If (itest2.GE.0) Then
         xstart=ncmin
         ystart=nrmin
         xsc=float(nxpix)
         ysc=float(nypix)
         Call setbnd
         Call plycom(itest)
      Else
         itest=-1
      End If
      If (ntop.GT.0)mdflag=1
      If (itest.GE.0)ifirst=0
C
      Return
      End
C+
C *************************************************************
C  plfout - Constructs a Plouff/Godson GRAVPOLY/MAGPOLY model
C           file from a POLYGON model.
C-
C *************************************************************
      Subroutine plfout(itest)
      Character ans*1,quest*80
C
      itest=0
      Do 10 while(itest.EQ.0)
         ans='G'
         quest=' Gravity or Magnetic model (g/m/h/q)'
         ival=iaquest(quest,ans,'(a2)',-2)
C
         If (ans.EQ.'G') Then
            Call grvmod(itest)
         Else If (ans.EQ.'M') Then
            Call magmod(itest)
         Else If (ans.EQ.'H') Then
```

```
            Call hlpplf
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C ****************************************************************
C  plycom - PoLY COMmand - Driver for Polygon mode.  Options
C           are.
C
C                a = Add a polygon
C                c = Change polygon parameters
C                d = Delete a polygon
C                e = Edit a polygon, (add/delete/move) points
C                s = Special functions (copy/move/rotate)
C                h = Help
C                q = Quit and return to Polygon Command level
C
C-
C ****************************************************************
      Subroutine plycom(itest)
      Character string*7,quest*80,ans*2
      string='ACDESHQ'
C
      Call enhmsg('*** POLYGON add/change_parm/delete/edit_poly mode
     & mode ***')
C
      Call clrply
C
      itest=0
      ans='h'
      Do 10 while(itest.EQ.0)
         iflag='P'
         If (iflag.EQ.'M') Then
            Call wrtmsg(' Polygon add/change_parm/delete/edit_poly mode
    & (a/c/d/e/s/h/q)')
            Call getmenu(string,7,ans,ierror)
         Else
            quest=' POLYGON add/change_parm/delete/edit_poly mode
    & (a/c/d/e/s/h/q)'
            ival=iaquest(quest,ans,'(a2)',2)
         End If
C
         If (ans.EQ.'A'.OR.ans.EQ.'a') Then
            Call addply(ierror)
         Else If (ans.EQ.'C'.OR.ans.EQ.'c') Then
            Call chgply(ierror)
         Else If (ans.EQ.'D'.OR.ans.EQ.'d') Then
            Call delply(ierror)
         Else If (ans.EQ.'E'.OR.ans.EQ.'e') Then
            Call edtply(ierror)
         Else If (ans.EQ.'S'.OR.ans.EQ.'s') Then
            Call spfcom(ierror)
         Else If (ans.EQ.'H'.OR.ans.EQ.'h') Then
            Call hlpply
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1.OR.ans.EQ.'q') Then
            itest=1
         Else
            Call errmsg
         End If
         ans='q'
   10 End Do
C
      Return
      End
C+
C **************************************************************
C  plytst - PoLY ToST - Tests whether a given location (x,y) is
C           inside (inout=1) or outside (inout=0) of the poly-
C           gon given by the arrays xpoly and ypoly.
C
C        Author: Richard J. Blakely, USGS, Menlo Park, CA.
C        Modified by Bruce A. Chuchel, USGS, Menlo Park, CA, 2/84.
C
C-
C **************************************************************
      Subroutine plytst(xpoly,ypoly,nbrpts,x,y,inout)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
C
      icount=0
      inout=0
      xmin=xpoly(1)
      xmax=xpoly(1)
      ymin=ypoly(1)
      ymax=ypoly(1)
C
      Do 10 j=1,nbrpts
      xmax=amax1(xpoly(j),xmax)
      xmin=amin1(xpoly(j),xmin)
      ymax=amax1(ypoly(j),ymax)
      ymin=amin1(ypoly(j),ymin)
   10 Continue
C
      If ((x.GE.xmin.AND.x.LE.xmax).AND.(y.GE.ymin.AND.y.LE.ymax)) Then
         Do 20 i=1,nbrpts
         iup=i+1
         if (i.EQ.nbrpts)iup=1
         x1max=amax1(xpoly(iup),xpoly(i))
         x1min=amin1(xpoly(iup),xpoly(i))
         If (x.GE.x1min AND.x.LT.x1max) Then
            a=(ypoly(iup)-ypoly(i))/(xpoly(iup)-xpoly(i))
            b=ypoly(i)-a*xpoly(i)
            y0=a*x+b
            If (y0.GE.y)icount=icount+1
         End If
   20    Continue
         inout=mod(icount,2)
      End If
C
      Return
      End
C+
C ***********************************************************************
C  rdeblank - Eliminates blanks at right end of string1, returns
C             as string2 with new length l2.
C
C     Author: Robert W. Simpson, USGS, Menlo Park, CA.
C-
C ***********************************************************************
      Subroutine rdeblank(string1,string2,l2)
      Character*(*) string1,string2
C
      l1=len(string1)
```

```
C  Find last non-blank character on right
      Do 10 i=l1,1,-1
   10 If (string1(i:i).NE.' ')Go To 20
   20 l2=i
      string2=string1
C
      Return
      End
C+
C ***************************************************************
C  rdhead - Reads in the header block of a standard grid (new and
C           old) versions.  If an error is encountered on reading
C           the header an attempt is made to read it as an old
C           standard grid.
C
C           itest = Flag for type of grid header read.
C                 = 1 = New version of standard (with proj,...)
C                 = 0 - Old version of standard (without proj,...)
C                 = -1= Error encountered, not a recognizable header
C
C  Author: Robert Simpson
C
C-
C ***************************************************************
      Subroutine rdhead(unit,id,pgm,ncol,nrow,nz,
     &xo,dx,yo,dy,iproj,cm,bl,itest)
      Character id*56,pgm*8
      Integer unit
      Read (unit,err=20)id,pgm,ncol,nrow,nz,xo,dx,yo,dy,iproj,cm,bl
C      print '(1h0,a,i2,2(a,f10.4))',
C     &    '  Proj=',iproj,'  cm=',cm,'  bl=',bl
      itest=1
      Return
C
   20 Continue
      Rewind unit
      Read (unit,err=30)id,pgm,ncol,nrow,nz,xo,dx,yo,dy
C      print '(1h0,a)', '  Grid has no projection specs in header...'
      iproj=0
      cm=0.0
      bl=0.0
      itest=0
      Return
C
   30 Continue
      Write (6,35)
   35 Format (/,' Error encountered on reading grid header',/)
      itest=-1
C
      Return
      End
C+
C ***************************************************************
C  readmd - READ MoDel - Reads in a Polygon model file.
C
C
C
C-
C ***************************************************************
      Subroutine readmd(itest)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100).vscr(100,100)
      Common /parameter /parm(100,10)
      Common /labels/label
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /names/grdnam,modnam,modgrd
      Common /commends/nmax,epsiln,delin,delout
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
C
      Character label(10)*15
      Character id*56,pgm*8,grdnam*80,modnam*80,modgrd*80
C
      Open (10,file=modnam,status='old',form='formatted',
     &carriagecontrol='list',err=100)
C
C - Read in the number of polygons in the model
C
      Read (10,10)ntotal
   10 Format (x,i3)
      Read (10,20)(label(kk),kk=1,5)
      Read (10,20)(label(kk),kk=6,10)
   20 Format (5(x,a15))
C
C - Read in the polygon number, the number of points in this
C   polygon, the screen locations and the polygon parms.
C
      xcndel=nxorgp/dx
      ycndel=nyorgp/dy
      icount=1
      Do 60 while((icount.GE.1).AND.(icount.LE.ntotal)
     &.AND.(icount.LE.nmax))
         Read (10,33)npoly,nbrpts
   33    Format (2(x,i3))
         Read (10,35)(parm(npoly,i),i=1,5)
         Read (10,35)(parm(npoly,i),i=6,10)
   35    Format (5(x,e15.8))
         numply(npoly)=nbrpts
C
         Do 40 j=1,nbrpts
         Read (10,45)xgrid,ygrid
   45    Format (2(x,e16.8))
         xscr(npoly,j)=iwcorg+((xgrid-xo)*xcndel)
         yscr(npoly,j)=jwcorg+((ygrid-yo)*ycndel)
   40    Continue
         icount=icount+1
   60 Continue
C
C - find garbage locations
C
      khit=0
      ngbloc=0
      Do 50 i=nmax,1,-1
      nloc=numply(i)
      If (nloc.GT.0.AND.khit.EQ.0) Then
         khit=1
      Else If (nloc.LE.0.AND.khit.EQ.1) Then
         ngbloc=ngbloc+1
         jnkloc(ngbloc)=i
      End If
   50 Continue
C
C - Now read in the topology structure
C
      ndpstn=1
      Do 70 k=1,nmax
      Read (10,65) info(k),iupper(k),idown(k),ileft(k),iright(k)
   65 Format (5(x,i3))
   70 Continue
C
C - Find the garbage topology locations.
C
      nhit=0
      ngbtop=0
      Do 80 j=nmax,1,-1
      inf=info(j)
      If (inf.GT.0.AND.nhit.EQ.0) Then
         nhit=1
      Else If (inf.LE.0.AND.nhit.EQ.1) Then
         ngbtop=ngbtop+1
         jnktop(ngbtop)=j
      End If
   80 Continue
C
      Close (10)
      itest=1
      Return
C
  100 Continue
      Close (10)
      itest=-1
      Return
      End
C+
C ***************************************************************
C    REDGRD - REaD GRid - Reads in a standard grid
C-
C ***************************************************************
      Subroutine redgrd(name,itest)
      Common /grid/grd(250000)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Character name*80,id*56,pgm*8
C
      itest=1
      Open (10,file=name,status='old',form='unformatted',err=100)
      Call rdhead(10,id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl,itest)
      If (itest.GE.0) Then
         Do 20 j=1,nr
         Read (10,End =20)dummy,(grd(i+(j-1)*nc),i=1,nc)
   20    Continue
      End If
      Close (10)
      Return
C
  100 Continue
      Close (10)
      itest=-1
      Return
      End
C+
C ***************************************************************
C  redwrk - Reads in a standard grid to the work array, called
C           by subroutine stdout (STANDARD OUT).
C-
C ***************************************************************
      Subroutine redwrk(name,itest)
      Common /work/wrkgrd(250000)
      Common /gridspecs2/id2,pgm2,nc2,nr2,nz2,xo2,dx2,yo2,dy2,
     &iproj2,cm2,bl2
      Character name*80,id2*56,pgm2*8
C
      itest=1
      Open (10,file=name,status='old',form='unformatted',err=100)
      Call rdhead(10,id2,pgm2,nc2,nr2,nz2,xo2,dx2,yo2,dy2,iproj2,cm2,
     &bl2,itest)
      If (itest.GE.0) Then
         Do 20 j=1,nr2
         Read (10,End =20)dummy,(wrkgrd(i+(j-1)*nc2),i=1,nc2)
   20    Continue
      End If
      Close (10)
      Return
C
  100 Continue
      Close (10)
      itest=-1
      Return
      End
C+
C ************************************************************
C  retpnt - RETurn PoINt - Returns the current Envision cross-
C           hair screen location (x,y), depending on whether the
C           mouse of cross-hair cursor keys were selected.
C
C              mcur = 'm' = Mouse is selected
C                     'c' = Cursor keys selected
C
C-
C ************************************************************
      Subroutine retpnt(x,y,mcur,itest)
      Character ans*1,mcur*2
C
      if (itest.eq.3) Call msgspm
C
C - If mouse was selected enable mouse and read in point.
C
      If (mcur.EQ.'M') Then
         Call softky('1')
         if (itest.eq.3.or.itest.eq.2) then
            Call setmou
            Call loadmou
         End If
         Call curon
         itest=0
         If (itest.EQ.0) Then
   20       Continue
            Call getmou(mode,ix,iy)
            x=float(ix)
            y=float(iy)
            If (mode.EQ.1) Then
               Call drwpnt(x,y)
               itest=1
            Else If (mode.EQ.2) Then
               Call drwpnt(x,y)
               itest=2
            Else If (mode.EQ.23) Then
               itest=-1
            Else If (mode.EQ.3) Then
               Call hlpmpt
            Else
               Call wrtmsg(' Mouse button not defined...try again')
            End If
            If (itest.EQ.0)Go To 20
         End If
         Call curoff
         Call softky('0')
```

```
C
C - Else if cursor was selected, enable the cross-hair cursors
C   and read in point.
C
      Else If (mcur.EQ.'C') Then
         Call curon
         itest=0
         If (itest.EQ.0) Then
   30       Continue
            Call gtpnt(ans,x,y)
            If (ans.EQ.'e') Then
               Call drwpnt(x,y)
               itest=1
            Else If (ans.EQ.'f') Then
               Call drwpnt(x,y)
               itest=2
            Else If (ans.EQ.'h') Then
               Call hlpcur
            Else If (ans.EQ.'q') Then
               itest=-1
            End If
            If (itest.EQ.0)Go To 30
         End If
         Call curoff
      End If
  100 Continue
C
      Return
      End
C+
C **************************************************************
C  rotate - Rotates the point xold,yold about the origin xorign,
C           yorign by the angle theta.
C-
C **************************************************************
      Subroutine rotate(xnew,ynew,xold,yold,xorign,yorign,theta)
C
      pi=3.14159265
      delx=xold-xorign
      dely=yold-yorign
      rlen=sqrt(delx**2+dely**2)
      If (rlen.LT.1.0e-16)rlen=1.0e-16
      alpha=asin(abs(delx/rlen))
      alpha=180.0*alpha/pi
C
      If (dely.GE.0.0) Then
         If (delx.LT.0.0)alpha=-alpha
      Else
         If (delx.GT.0.0) Then
            alpha=180.0-alpha
         Else
            alpha=alpha+180.0
         End If
      End If
      alpha=90.0-alpha
      If (alpha.LT.0.0)alpha=360.0+amod(alpha,360.0)
C
      xnew=rlen*cos((theta+alpha)*pi/180.0)+xorign
      ynew=rlen*sin((theta+alpha)*pi/180.0)+yorign
C
      Return
      End
C+
C **************************************************************
```

```
C  rotply - Allows a polygon to be rotated about a selected origin
C           by the amount theta.
C-
C ****************************************************************
      Subroutine rotply(itest)
      Dimension perstore(10)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /parameter /parm(100,10)
      Common /polyloc/nptloc,xloc(100),yloc(100)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /flags/mcflag,votflg
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /gridspecs/ id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /scale/ xsc,ysc,xstart,ystart,xinit,yinit
      Common /original/ iwcorg,jwcorg,nxorgp,nyorgp
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /state/lflast,lftin,lftout,ndpstn,iup
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character votflg*2,quest*80,mcflag*2,mcur*2,ans*2,ans2*2
      Character id*56,pgm*8,ans3*2,ans4*2,use*2
C
      Call enhmsg('*** Rotate polygon mode ***')
C
      itest=0
C
C - Test the number of polygons ntotal, exit if <=0.
C
      If (ntotal.GT.0) Then
C
C - If cursor type has not been selected prompt for type.
C
         If (mcflag.EQ.'N') Then
            Call askmoc(mcur)
            If (mcur.EQ.'Q') Then
               itest=-1
            Else
               mcflag=mcur
            End If
         Else
            mcur=mcflag
         End If
C
C - Print help message
C
         If (votflg.EQ.'V')Call hlprot
C
C - Start looping until polygon is found (itest=1), or user
C   wants to quit (itest=-1).
C
         Do 10 while(itest.EQ.0)
         Iflag=0
         If (mcur.EQ.'M'.OR.mcur.EQ.'C') Then
            use='M'
C
C - Initialize the temp arrays.
C
```

```
            Call inttmp
C
C - Let user pick polygon, and return npoly,ncorn,x,y.
C
            Call pckply(npoly,ncorn,x,y,ans,mcur,ierr)
C
C - Ask If corner of polygon picked should be used
C
            If (ans.EQ.'Y') Then
               nbrpts=numply(npoly)
               Call askrot(ans2)
               If (ans2.EQ.'Y') Then
                  x0=xscr(npoly,ncorn)
                  y0=yscr(npoly,ncorn)
                  use='Y'
               Else If (ans2.EQ.'N') Then
C
C - Message about repositioning cursor to new location
C
                  Call askrt2(ans3)
                  If (ans3.EQ.'Y') Then
                     Call modpnt(ncorn,dist,x,y,ans4,npoly,mcur)
                     If (ans4.EQ.'Y') Then
                        x0=xscr(npoly,ncorn)
                        y0=yscr(npoly,ncorn)
                        use='Y'
                     Else
                        itest=-2
                     End if
                  Else If (ans3.EQ.'N') Then
                     Call hlprt2
                     Call retpnt(xscnew,yscnew,mcur,ierr)
                     If (ierr.GE.0) Then
                        If (izoom.EQ.1) Then
                           Call invers(xnew,ynew,xscnew,yscnew)
                        Else
                           xnew=xscnew
                           ynew=yscnew
                        End If
                        x0=xnew
                        y0=ynew
                     Else
                        itest=-1
                     End If
                  Else If (ans3.EQ.'Q') Then
                     itest=-2
                  End If
               Else If (ans2.EQ.'Q') Then
                  itest=-2
               End If
               If (itest.LE.-1)Go To 20
C
C - User enters rotation angle in degrees counter-clockwise.
C
               itest4=0
               Do 45 while(itest4.EQ.0)
               Write (6,40)
  40           Format (/,
    &          '  You will now enter the rotation angle for',
    &          ' this polygon, measured',/,' in degrees ',
    &          ' counter-clockwise.',/)
               theta=0.0
               quest=' Rotation angle'
               ival=irquest(quest,theta,'(e16.8)',0)
```

```
C
               If (ival.EQ.-1) Then
                  itest4=-2
                  itest=-2
               Else
                  itest4=1
               End If
   45          End Do
            If (itest.LE.-1)Go To 20
C
C - Rotate polygon and store.
C
            ntemp=nbrpts
            Do 30 i=1,ntemp
            If (use.EQ.'Y'.AND.i.EQ.ncorn) Then
               xtemp(i)=xscr(npoly,ncorn)
               ytemp(i)=yscr(npoly,ncorn)
            Else
               Call rotate(xtemp(i),ytemp(i),xscr(npoly,i),
     & yscr(npoly,i),x0,y0,theta)
            End If
   30       Continue
C
C - Test the polygon to see if it has been rotated completely off
C   the unzoomed screen.
C
            xlfunz=xinit
            xrgunz=xinit+nc*nxorgp
            ybtunz=yinit
            ytpunz=yinit+nr*nyorgp
            Call testoff(xtemp,ytemp,ntemp,xlfunz,xrgunz,ybtunz,
     &      ytpunz,intotal)
            If (intotal.LE.0) Then
               If (intotal.EQ.0) Call wrtmsg(' Error, polygon will be
     & rotated off of unzoomed screen')
               Go To 20
            End If
C
C - Store the polygon parameters.
C
            Do 25 ii=1,10
            parstore(ii)=parm(npoly,ii)
   25       Continue
C
C - Test the new polygon in the xtemp,ytemp array and fit into
C   the topology structure.
C
            Call testopo(npoly,itest2)
            If (itest2.GE.1) Then
               npoly2=info(ndpstn)
               Do 35 iii=1,10
               parm(npoly2,iii)=parstore(iii)
   35          Continue
               Iflag=1
            End If
            itest=1
C
   20       Continue
            If (Iflag.EQ.0) Then
               Call setclr(plyclr)
               Call setfil(open)
               Call drwclp(xloc,yloc,nptloc,xleft,xright,ybot,ytop)
            End If
         Else If (ans.EQ.'N') Then
```

```
            itest=0
         Else
            itest=-1
         End If
         Call setfil(solid)
         If (ierr.EQ.-1)itest=-1
      Else If (mcur.EQ.'Q') Then
         itest=-1
      End If
   10 End Do
      Else
         Call wrtmsg('  Sorry, no polygons')
         itest=-1
      End If
C
      Return
      End
C+
C **************************************************************
C  rstply - ReSeT PoLY - Resets grid locations in wrkgrd under
C           control of irst.  If irst=1, resets grid points
C           within polygon xtmp,ytmp; if irst=0, resets grid
C           points outside of polygon xpoly,ypoly.
C-
C **************************************************************
      Subroutine rstply(xpoly,ypoly,nbrpts,ncmin,ncmax,nrmin,nrmax,
     &nc,nr,rstval,irst,itest)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
      Common /work/wrkgrd(250000)
C
      If ((nrmin.LE.nrmax).AND.(nrmin.GE.1.AND.nrmin.LE.nr).AND.
     &(nrmax.GE.1.AND.nrmax.LE.nr)) Then
         itest=1
      Else
         Print *,' Error in nrmin and nrmax'
         Print *,' nrmin=',nrmin,' nrmax=',nrmax
         itest=-1
      End If
C
      If ((ncmin.LE.ncmax).AND.(ncmin.GE.1.AND.ncmin.LE.nc).AND.
     &(ncmax.GE.1.AND.ncmax.LE.nc)) Then
         itest=1
      Else
         Print *,' Error in ncmin and ncmax'
         Print *,' ncmin=',ncmin,' ncmax=',ncmax
         itest=-1
      End If
      If (itest.EQ.-1)Go To 50
C
      inout=1
      Do 50 j=nrmin,nrmax
      Do 50 i=ncmin,ncmax
      xi=i
      yi=j
C
      Call plytst(xpoly,ypoly,nbrpts,xi,yi,inout)
C
      If (inout.EQ.irst) Then
         wrkgrd(i+(j-1)*nc)=rstval
      End If
   50 Continue
C
      Return
      End
```

```
C+
C ***********************************************************************
C - Scales grid to fill screen...
C   Screen is (4151x3120) in world coordinates wc = (iwc,jwc).
C   A pixel here is a box around 1 wc point = smallest possible box.
C   A box is an (nxpix X nypix) set of pixels.
C   Define area of screen for drawing...
C
C-
C *************************************************************
      Subroutine scaleg2sc(iflag)
      Character esc*1,wcbp*5,id*56,pgm*8
      Parameter (esc=char(27))
      Common /scale/ xsc,ysc,xstart,ystart,xinit,yinit
      Common /subscreen/ xscrn(2),yscrn(2),xgrd(2),ygrd(2)
      Common /gridspecs/id,pgm,ncol,nrow,nz,xo,dx,yo,dy
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
C
      iwcmin=nint(xinit)
      jwcmin=nint(yinit)
      iwcmax=nint(xscrn(2))
      jwcmax=nint(yscrn(2))
C
      ncdiff=ncmax-ncmin+1
      nrdiff=nrmax-nrmin+1
C - Width and ht of grid in grid units (usually km)...
      If (ncdiff.EQ.0)ncdiff=1
      If (nrdiff.EQ.0)nrdiff=1
      width=ncdiff*dx
      ht=nrdiff*dy
C - Dimension of allowed screen area in pixels...
      iscr=iwcmax-iwcmin+1
      jscr=jwcmax-jwcmin+1
C - Get scale factors (km/pixel) for both x and y directions...
C   Choose that factor (the bigger) which will just fit grid to screen
      pixdim=max(width/real(iscr),ht/real(jscr))
      nxtotpix=width/pixdim
      nytotpix=ht/pixdim
C - Find pixel dimensions of a box by rounding...
      nxpix=nint((width/pixdim)/real(ncdiff))
      nypix=nint((ht/pixdim)/real(nrdiff))
C - Check result against screen size...if too big cut back by 1...
  200 Continue
      If (nxpix*ncdiff.GT.iscr) Then
         nxpix=nxpix-1
         Go To 200
      End If
C
  210 Continue
      If (nypix*nrdiff.GT.jscr) Then
         nypix=nypix-1
         Go To 210
      End If
C - Check final scaling...
      wbyh=real(nxpix*ncdiff)/real(nypix*nrdiff)
      wbyhtrue=width/ht
      fracerr=(wbyh-wbyhtrue)/wbyhtrue
C     If (iflag.EQ.1) Then
C        Print *,' '
C        Print *,' x vs. y scale distortion =',fracerr
C        Print *,wbyh,wbyhtrue
C     End If
      If (abs(fracerr.GT.0.05)) Then
```

```
         Print *,'WARNING...grid ht vs. width distorted by ',fracerr
         Print *,' Grid will probably scale better if it is coarser...'
      End If
C
C - Center of first box, and origin of grid coordinate system...
C   (a box is centered on each grid point)
C
      iwc0=iwcmin+nxpix/2
      jwc0=jwcmin+nypix/2
C     If (iflag.EQ.1) Then
C        Print *,' '
C        Print *,
C    &   'ncol,nrow,nxtotpix,nytotpix,nxpix,nypix'
C        Print *,
C    &   ncol,nrow,nxtotpix,nytotpix,nxpix,nypix
C     End If
C
      Return
      End
C+
C *******************************************************************
C  scalen - Determines the scaling factors XSC,YSC for a (sub)grid.
C-
C *******************************************************************
      Subroutine scalen
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
C
      xsc=0.0
      ysc=0.0
C
      If ((ncmax-ncmin).GT.1)xsc=1.0*nxpix
      If ((nrmax-nrmin).GT.1)ysc=1.0*nypix
C
      Return
      End
C+
C ************************************************************
C  selftest - Tests whether a polygon is self-crossing (self-
C             reentrant).  Test by comparing the sides in pairs for
C             crossing.  Subroutine exits immediately on finding
C             the first pair of sides (NSIDE1,NSIDE2) that do cross.
C
C     FORM.
C
C        CALL SELFTEST(IFLAG,NSIDE1,NSIDE2,XPOLY,YPOLY,NBRPTS)
C
C            iflag = Flag indicating whether a polygon is self-
C                    crossing or not.
C                  = 1 - polygon is NOT self-crossing
C                  = 0 - polygon passed has three corners or less.
C                  =-1 - polygon is self-crossing.
C
C   nside1,nside2 = Sides of polygon that cross; returned nonzero
C                   when iflag=-1.  NSIDE1 is defined as the line
C                   segment joining the corner point xpoly(nside1),
C                   ypoly(nside1) to xpoly(nside1+1),ypoly(nside1+1).
C
C     xpoly,ypoly = Array containing corner points of polygon
C
C          nbrpts = Number of corner points in polygon xpoly.ypoly
C
C     Author: Bruce A. Chuchel, USGS, Menlo Park, CA., 1/85.
```

```
C-
C **************************************************************
      Subroutine selftest(iflag,nside1,nside2,xpoly,ypoly,nbrpts)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
C
C - Test to see if the polygon passed to SELFTEST has the minimum
C   number of points (4).
C
      If (nbrpts.GE.4) Then
C
C - Loop through sides and test for crossing.
C
         ilow=1
         Do 20 i=3,nbrpts
         If (i.EQ.nbrpts) Then
            x1=xpoly(nbrpts)
            y1=ypoly(nbrpts)
            x2=xpoly(1)
            y2=ypoly(1)
            ilow=2
         Else
            x1=xpoly(i)
            y1=ypoly(i)
            x2=xpoly(i+1)
            y2=ypoly(i+1)
         End If
C
C - Inner loop points to only those sides in the polygon that
C   are before side i, and are not adjacent to it.
C
         Do 20 j=ilow,i-2
         If (j.EQ.nbrpts) Then
            xt1=xpoly(nbrpts)
            yt1=ypoly(nbrpts)
            xt2=xpoly(1)
            yt2=ypoly(1)
         Else
            xt1=xpoly(j)
            yt1=ypoly(j)
            xt2=xpoly(j+1)
            yt2=ypoly(j+1)
         End If
C
C - Test corner points of individual line segments i and j for crossing.
C
         Call tstseg(ncross,x1,y1,x2,y2,xt1,yt1,xt2,yt2)
C
C - If line segments cross then, set iflag, nside*, and exit.
C
         If (ncross.EQ.1) Then
            iflag=-1
            nside1=i
            nside2=j
            Go To 100
         End If
   20    Continue
         iflag=1
         nside1=0
         nside2=0
C
      Else
         iflag=0
         nside1=0
         nside2=0
      End If
C
  100 Continue
      Return
      End
C+
C ****************************************************************
C  setbnd - Sets up the screen boundary common block (xleft,xright,
C           ybot,ytop).
C-
C ****************************************************************
      Subroutine setbnd
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /scalofacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Character id*56,pgm*8
C
      xleft=xinit
      ybot=yinit
C
      If (izoom.EQ.1) Then
         xright=(ncmax-ncmin+1)*nxpix+xinit
         ytop=(nrmax-nrmin+1)*nypix+yinit
      Else
         xright=nc*nxorgp+xinit
         ytop=nr*nyorgp+yinit
      End If
C
      Return
      End
C+
C ****************************************************************
C  setclr - Sets the graphics foreground color.  See setclr.inf
C           or Envision reference manual for parameters.
C-
C ****************************************************************
      Subroutine setclr(color)
      Character com*2,color*1
C
      com='C'//color
      Call esccom(com)
C
      Return
      End
C+
C ****************************************************************
C  setdlm - Sets the drawing logic mode on the Envision.  See the
C           Envision reference manual or setdlm.inf for details.
C-
C ****************************************************************
      Subroutine setdlm(mode)
      Character mode*1,com*2
C
      com='L'//mode
      Call esccom(com)
C
      Return
      End
C+
C ****************************************************************
C - setfil - SET FILl - Select the fill pattern.
C-
C ****************************************************************
      Subroutine setfil(fill)
      Character fill*1,com*3
C
      com='OH'//fill
      Call esccom(com)
C
      Return
      End
C+
C ****************************************************************
C  setgrd - Walks the tree starting at the root node and sets
C           the grid.
C-
C ****************************************************************
      Subroutine setgrd(ival,itest)
      Dimension xpoly(100),ypoly(100)
      Common /subscreen/xscrn(2),yscrn(2),xgrd(2),ygrd(2)
      Common /topology/info(100),iupper(100),idown(100),ileft(100),
     &iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /max/nptmax
      Common /parameter /parm(100,10)
      Common /work/wrkgrd(250000)
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
C
      Character id*56,pgm*8
C
      Call fndtop(ntop)
      If (ntop.GE.1) Then
C
         next=ntop
         rstval=0.0
         itest=0
C
         Do 5 j=1,nptmax
         xpoly(j)=0.0
         ypoly(j)=0.0
    5    Continue
C
         rscxsc=0.0
         rscysc=0.0
         If (abs(xsc).GT.1.e-16)rscxsc=1.0/xsc
         If (abs(ysc).GT.1.e-16)rscysc=1.0/ysc
C
         Do 10 while(itest.EQ.0.AND.next.GT.0)
            ngon=next
            npoly=info(ngon)
            nbrpts=numply(npoly)
            If (nbrpts.GE.1) Then
               Do 30 i=1,nbrpts
               xpoly(i)=xstart+(xscr(npoly,i)-iwcorg)*rscxsc
               ypoly(i)=ystart+(yscr(npoly,i)-jwcorg)*rscysc
   30          Continue
C
               rstval=parm(npoly,ival)
C
```

```
C - Call up the bounding box and translate these coordinates into
C   subgrid locations (ncmin,ncmax,nrmin,nrmax).
C
               xgrd(1)=0.0
               xgrd(2)=5.0
               ygrd(1)=0.0
               ygrd(2)=0.0
C
               xscrn(1)=xminbx(npoly)
               xscrn(2)=xmaxbx(npoly)
               yscrn(1)=yminbx(npoly)
               yscrn(2)=ymaxbx(npoly)
C
               Do 35 j=1,2
               xgrd(j)=xstart+(xscrn(j)-iwcorg)*rscxsc
               ygrd(j)=ystart+(yscrn(j)-jwcorg)*rscysc
   35          Continue
C
               ncmin=iint(xgrd(1))
               ncmax=iint(xgrd(2))+1
               nrmin=iint(ygrd(1))
               nrmax=iint(ygrd(2))+1
               If (ncmin.LT.1)ncmin=1
               If (ncmax.GT.nc)ncmax=nc
               If (nrmin.LT.1)nrmin=1
               If (nrmax.GT.nr)nrmax=nr
C
C
               irst=1
               Call rstply(xpoly,ypoly,nbrpts,ncmin,ncmax,nrmin,nrmax,
     &      nc,nr,rstval,irst,iflag)
C
            End If
            Call walk(next,ngon,itest2)
            If (ncxt.EQ.0.OR.itest2.EQ.0)itest=1
            If (itest2.EQ.-1)itest=-1
   10    End Do
      End If
C
      Return
      End
C+
C ************************************************************
C
C  setkam - SET Key Application Mode - Sets the Envision term-
C           inal up in key pad application mode.
C
C-
C ************************************************************
      Subroutine setkam
C
      Call esccom('=')
C
      Return
      End
C+
C ************************************************************
C - setlin - SET LINe - Select the line style.
C-
C ************************************************************
      Subroutine setlin(line)
      Character line*1,com*2
C
      com='T'//line
      Call esccom(com)
C
      Return
      End
C+
C ****************************************************************
C  setmou - Sets up the mouse soft key button definitions, called
C           by subroutine getmou (GET MOUSE).
C           Author: Robert W. Simpson
C-
C ****************************************************************
      Subroutine setmou
      Character esc*1,alphup*8,alphoff*8,alphon*3
      Parameter (esc=char(27))
      Parameter (alphup='\<'//esc//'[A'//'\^?')
      Parameter (alphoff='\<'//esc//'a0'//'\^?')
      Parameter (alphon=esc//'a2')
C
      Call esccom('[A'//esc//'RD\1*'//alphoff//alphup//'  3\_\''*')
      Call esccom('[A'//esc//'RD\2*'//alphoff//alphup//'  2\_\''*')
      Call esccom('[A'//esc//'RD\3*'//alphoff//alphup//' 23\_\''*')
      Call esccom('[A'//esc//'RD\4*'//alphoff//alphup//'  1\_\''*')
      Call esccom('[A'//esc//'RD\5*'//alphoff//alphup//' 13\_\''*')
      Call esccom('[A'//esc//'RD\6*'//alphoff//alphup//' 12\_\''*')
      Call esccom('[A'//esc//'RD\7*'//alphoff//alphup//'123\_\''*')
C
      Return
      End
c+
c ***************************************************************
c  settpl - Sets the pointers in the newtopo arrays for the case
c           when there are multiple polygons, both inside and
c           outside of npoly, at the same level in the tree.
c
c            in - Array containing node positions of all the polygons
c                 contained in npoly.
c
c          jout - Array of node positions of all those polygons
c                 not contained in npoly, but at the same
c                 level in the tree as npoly; jout is constructed
c                 if an element in the IN array exists.
c
c         lftin - Node position of first polygon in IN array that
c                 polygon npoly contains.
c
c        lftout - The left most node
c-
c ***************************************************************
      Subroutine settpl(npoly)
      Common /newtopo/infnew(100),iupnew(100),idwnew(100),
     &ilfnew(100),irtnew(100)
      Common /neighbors/icnt,in(100),jcnt,jout(100)
      Common /state/lflast,lftin,lftout,ndpstn,iup
C
c - Start by moving the left and right pointers of all the
c   "neighbors" of npoly in the tree.
C
      If (jcnt.GT.0) Then
         If (lflast.eq.0.and.iup.gt.0) idwnew(iup)=lftout
         Do 35 j=1,jcnt
         If (j.EQ.1) Then
            ilfnew(lftout)=lflast
         Else
            ilfnew(jout(j))=jout(j-1)
         End If
C
         If (j.EQ.jcnt) Then
            irtnew(jout(jcnt))=ndpstn
            ilfnew(ndpstn)=jout(jcnt)
         Else
            irtnew(jout(j))=jout(j+1)
         End If
   35    Continue
         ilfnew(ndpstn)=jout(jcnt)
         If (lflast.gt.0) irtnew(lflast)=lftout
      Else If (jcnt.eq.0) Then
         ilfnew(ndpstn)=lflast
         If (lflast.gt.0) irtnew(lflast)=ndpstn
         If (lflast.eq.0.and.iup.gt.0) idwnew(iup)=ndpstn
      End If
C
      infnew(ndpstn)=npoly
      iupnew(ndpstn)=iup
      idwnew(ndpstn)=lftin
      irtnew(ndpstn)=0
C
c - Move the up, left, and right pointers of those objects now
c   contained in polygon npoly (i.e. have ndpstn as a parent).
C
      if (icnt.gt.0) then
         Do 40 i=1,icnt
         iupnew(in(i))=ndpstn
         If (i.EQ.1) Then
            ilfnew(lftin)=0
         Else
            ilfnew(in(i))=in(i-1)
         End If
C
         If (i.EQ.icnt) Then
            irtnew(in(i))=0
         Else
            irtnew(in(i))=in(i+1)
         End If
   40    Continue
      End If
C
      Return
      End
C+
C ***************************************************************
C  softky - SOFT KeY - Sets the sending of soft key definitions
C           ('0'=disable,'1'=enable).
C-
C ***************************************************************
      Subroutine softky(comd)
      Character com*3,comd*1
C
      com='Rm'//comd
      Call esccom(com)
C
      Return
      End
C+
C **************************************************************
C  spfcom - Driver for special function mode. Options are:
C
C
C              c = Copy a polygon (and parameters)
```

```
C              m = Move a polygon to a new position)
C              r = Rotate a polygon about a specified origin and
C                    by a prescribed theta.
C              h = Help
C              q = Quit and return to polygon ADD/CHANGE/DELETE/
C                    EDIT mode.
C
C-
C *************************************************************
      Subroutine spfcom(itest)
      Character quest*80,ans*2
C
      Call enhmsg('*** Special functions mode ***')
C
C
      ans='h'
      itest=0
      Do 10 while(itest.EQ.0)
         quest='  Special function mode (c/m/r/h/q)'
         ival=iaquest(quest,ans,'(a2)',2)
C
         If (ans.EQ.'C'.OR.ans.EQ.'c') Then
            Call cpyply(ierror)
         Else If (ans.EQ.'M'.OR.ans.EQ.'m') Then
            Call movply(ierror)
         Else If (ans.EQ.'R'.OR.ans.EQ.'r') Then
            Call rotply(ierror)
         Else If (ans.EQ.'H'.OR.ans.EQ.'h') Then
            Call hlpspf
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1.OR.ans.EQ.'q') Then
            itest=1
         Else
            Call errmsg
         End If
         ans='q'
   10 End Do
C
      Return
      End
C+
C ************************************************************
C  stdout - Creates a Denver Standard grid output file.
C-
C ************************************************************
      Subroutine stdout(itest)
      Common /gridspecs2/id2,pgm2,nc2,nr2,nz2,xo2,dx2,yo2,dy2,
     &iproj2,cm2,bl2
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /names/grdnam,modnam,modgrd
      Character ulabel*15,quest*80,grdnam*80,modnam*80,modgrd*80
      Character id2*56,pgm2*8,ans*2,iddum*80,id*56,pgm*8
C
C - Ask if user want to use an already existing grid.
C
      iflag=1
      Call msgstd
      quest=' Do you wish to use a prexisiting grid (y/n/q)'
      Call askqst(ans,quest,itest)
      If (itest.EQ.-1.OR.ans.EQ.'Q')Go To 100
      If (ans.EQ.'Y')iflag=2
C
C - Ask for model grid's name
C
      quest=' Model grid'
      Call asknam(modgrd,quest,itest)
      If (itest.EQ.-1)Go To 100
C
C - If a prexisting grid is used read it in and test its grid specs
C
      If (iflag.EQ.2) Then
         Call redwrk(modgrd,itest)
         Call testspecs(itest)
         If (itest.LE.-1)Go To 100
      Else
         Call copyspecs
      End If
C
C - ASK FOR GRID ID.
C
      quest=' Id'
      Call asknam(iddum,quest,itest)
      If (itest.EQ.-1)Go To 100
      ileng=itlen(iddum)
      id2=iddum(1:56)
C
C - Ask for grid dval
C
      If (iflag.EQ.1)Call askdvl(dval,itest)
      If (itest.EQ.-1)Go To 100
C
C - Ask for which set of parameters (PARM* arrays) to use in
C   setting the grid
C
      Call askval(iset,ulabel,itest)
      If (itest.EQ.-1)Go To 100
C
C - Initialize wrkgrd to dval
C
      If (iflag.EQ.1)Call intwrk(dval)
C
C - Walk the tree and set the grid
C
      Call setgrd(iset,itest)
C
C - Write out the grid to the user specified file
C
      Call wrtgrd(modgrd,itest)
C
  100 Continue
      Return
      End
C+
C ************************************************************
C  stoply - Stores the polygon xpoly,ypoly in the screenloc array
C           at position npoly.
C-
C ************************************************************
      Subroutine stoply(xpoly,ypoly,nbrpts,npoly,itest)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
C
      numply(npoly)=nbrpts
      Do 10 i=1,nbrpts
      xscr(npoly,i)=xpoly(i)
      yscr(npoly,i)=ypoly(i)
   10 Continue
C
      Return
      End
C+
C *************************************************************
C  storeold - Stores all the polygon information associated with
C              npoly in the oldcoord common block.
C-
C *************************************************************
      Subroutine storeold(npoly,itest)
      Common /oldcoord/nold,xold(100),yold(100),xinold(100),
     &yinold(100),xoutold(100),youtold(100),xminold,xmaxold,yminold,
     &ymaxold,partmp(10)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /inout/xin(100,100),yin(100,100),xout(100,100),
     &yout(100,100)
      Common /parameter /parm(100,10)
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Then
         nold=numply(npoly)
         If (nold.GE.1.AND.nold.LE.nptmax) Then
            xminold=xminbx(npoly)
            xmaxold=xmaxbx(npoly)
            yminold=yminbx(npoly)
            ymaxold=ymaxbx(npoly)
            Do 10 i=1,nold
            xold(i)=xscr(npoly,i)
            yold(i)=yscr(npoly,i)
            xinold(i)=xin(npoly,i)
            yinold(i)=yin(npoly,i)
            xoutold(i)=xout(npoly,i)
            youtold(i)=yout(npoly,i)
   10       Continue
            Do 20 j=1,10
            partmp(j)=parm(npoly,j)
   20       Continue
            itest=1
         Else
            Print *,' Error, number of points for polygon',npoly,
     &      ' out of range'
            itest=-1
         End If
      Else
         Call wrtmsg(' Polygon number out of range')
         itest=-1
      End If
C
      Return
      End
C+
C *************************************************************
C  tangle - Turning ANGLE
C
C    PURPOSE.
C       Determines the turning angle (relative heading change)
C           THETAD in degrees when going from the line segment
C           (x1,y1)-(x2,y2) to the line segment (x2,y2)-(x3,y3).
C
C           Using the equation:
C
C              c**2=a**2+b**2-2.0*a*b*cos(alpha)
C
C           where a,b are the distances from the corner (x1,y1),
C           (x2,y2) and (x2,y2),(x3,y3) respectivaly; and the
C           turning angle is defined as:
C
C              thetad=180.0-alpha
C
C    FORM.
C
C           Call tangle(thetad,x1,y1,x2,y2,x3,y3)
C
C    PARAMETERS.
C
C           THETAD - Turning angle in degrees.
C
C    (x1,y1),(x2,y2),(x3,y3) - End points of line segments.
C
C    NOTE:  The THETAD returned is a positive number between 0.0
C           and 180.0 degrees; TANGLE does not make a distinction
C           between a left or right turn.
C-
C ***************************************************************
      Subroutine tangle(thetad,x1,y1,x2,y2,x3,y3)
C
      a=sqrt((x2-x1)**2+(y2-y1)**2)
      b=sqrt((x3-x2)**2+(y3-y2)**2)
      c=sqrt((x3-x1)**2+(y3-y1)**2)
C
      If (a.LT.1.0e-18)a=1.0e-18
      If (b.LT.1.0e-10)b=1.0e-10
C
      thetad=acosd((c**2-(a**2+b**2))/(2.0*a*b))
C
      Return
      End
C+
C ****************************************************************
C  tester - Starting at polygon (LPOLY), tester walks the tree
C           structure and returns the polygon number (NPOLY) and
C           the corner of this polygon (NCORN) closest to the
C           point (x,y).
C-
C ****************************************************************
      Subroutine tester(npoly,ncorn,lpoly,x,y,itest)
      Dimension xpoly(100),ypoly(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Parameter (vaxmin=-1.7e+38,vaxmax=1.7e+38)
      Character lowup*3
C
      npoly=0
      ncorn=0
      If (lpoly.GE.1.AND.lpoly.LE.nmax) Then
         itest=0
         ipoly=lpoly
         icorn=0
         icorn2=0
         iset=0
         iset2=0
         npoly1=0
         ncorn1=0
         npoly2=0
         ncorn2=0
      dmin=vaxmax
      dmin2=vaxmax
C
      Do 10 while(itest.EQ.0)
      Call tstbrd(inout,x,y,ipoly)
      If (inout.EQ.1) Then
        nbrpts=numply(ipoly)
        If (nbrpts.GE.1.AND.nbrpts.LE.nptmax) Then
          Do 30 i=1,nbrpts
          xpoly(i)=xscr(ipoly,i)
          ypoly(i)=yscr(ipoly,i)
  30      Continue
C
C - Test the polygon by using both the limited and unlimited
C   versions of subroutine clspnt.
C
          iflag=0
          Call clspnt(icorn,dist,x,y,xpoly,ypoly,nbrpts,delout,
     &    iflag)
          iflag2=1
          Call clspnt(icorn2,dist2,x,y,xpoly,ypoly,nbrpts,
     &    delout,iflag2)
          If (icorn.GT.0.AND.dist.LT.dmin) Then
            npoly1=ipoly
            ncorn1=icorn
            dmin=dist
            iset=1
          End If
          If (icorn2.GT.0.AND.dist2.LT.dmin2) Then
            npoly2=ipoly
            ncorn2=icorn2
            dmin2=dist2
            iset2=1
          End If
C
C - If the point (x,y) was not within the radius used for
C   the limited version of clspnt, then test to see if the point
C   is inside of the polygon currently under test.
C
          If (icorn2.EQ.0) Then
            Call plytst(xpoly,ypoly,nbrpts,x,y,inout2)
            Call fndply(ngon,ipoly)
            If (inout2.EQ.1) Then
              next=idown(ngon)
            Else
              next=iright(ngon)
            End If
            If (next.LE.0) Then
              If (iset2.EQ.1) Then
                npoly=npoly2
                ncorn=ncorn2
                itest=1
              Else If (iset.EQ.1) Then
                npoly=npoly1
                ncorn=ncorn1
                itest=1
              Else
                itest=-1
              End If
            Else
              ipoly=info(next)
            End If
          Else If (icorn.GT.0.OR.icorn2.GT.0) Then
            Call fndply(ngon,ipoly)
```

```
                  If (ngon.GE.1) Then
                     Call walk(next,ngon,itest3)
                     If (next.EQ.0.OR.itest3.EQ.0) Then
                        If (iset2.EQ.1) Then
                           npoly=npoly2
                           ncorn=ncorn2
                           itest=1
                        Else If (iset.EQ.1) Then
                           npoly=npoly1
                           ncorn=ncorn1
                           itest=1
                        Else
                           Print *,' '
                           itest=-1
                        End If
                     Else
                        ipoly=info(next)
                     End If
                  Else
                     Print *,' Error, ngon not found'
                     itest=-1
                  End If
               Else
                  Print *,' Error encountered in clspnt at polygon',
     &            ipoly,' and corner # ',icorn
                  itest=-1
               End If
            Else
               Print *,' Error in polygon ',ipoly,
     &         ' nbrpts out of range'
               itest=-1
            End If
         Else If (inout.EQ.0) Then
            Call fndply(ngon,ipoly)
            next=iright(ngon)
            If (next.LE.0) Then
               If (iset2.EQ.1) Then
                  npoly=npoly2
                  ncorn=ncorn2
                  itest=1
               Else If (iset.EQ.1) Then
                  npoly=npoly1
                  ncorn=ncorn1
                  itest=1
               Else
                  itest=-1
                  Print *,' Test bound routine'
                  Print *,' '
C                    print *,' Sorry, could not find your polygon'
               End If
            Else
               ipoly=info(next)
            End If
         Else
            Print *,' Error encountered in tstbnd at polygon',ipoly
            itest=-1
         End If
   10    End Do
      Else
         Print *,' Error, lpoly=',lpoly,' passed to tester out of range'
         itest=-1
      End If
C
      Return
      End
C+
C ************************************************************
C  testoff - Tests the polygon xpoly,ypoly to see if at least one
C            corner point of the polygon is inside or outside of a
C            given window.
C
C            Window is defined as x,y:
C               xleft=<x=<xright
C               ybot=<y=<ytop
C
C            intotal = Flag giving result of test.
C                    >= 1, at least one point is within window
C                     = 0, polygon is outside of window
C                     =-1, error in parameters passed to routine.
C-
C ************************************************************
      Subroutine testoff(xpoly,ypoly,nbrpts,xleft,xright,ybot,
     &ytop,intotal)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
C
      If ((nbrpts.GE.1).AND.(xleft.LE.xright).AND.(ybot.LE.ytop)) Then
         intotal=0
         Do 10 i=1,nbrpts
         If ((xpoly(i).GE.xleft.AND.xpoly(i).LE.xright).AND.
     &   (ypoly(i).GE.ybot.AND.ypoly(i).LE.ytop))intotal=intotal+1
   10    Continue
      Else
         Call wrtmsg(' Error in TESTOFF, boundary conditions or nbrpts
     &   illegal')
         intotal=-1
      End If
C
      Return
      End
C+
C ************************************************************
C  testopo - Test the polygon passed in the xtemp,ytemp to see if
C            it will fit into the topology arrays.  If not, restores
C            the old polygon information stored in the oldcoord
C            common block.
C-
C ************************************************************
      Subroutine testopo(npoly,itest)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /oldcoord/nold,xold(100),yold(100),xinold(100),
     &yinold(100),xoutold(100),youtold(100),xminold,xmaxold,yminold,
     &ymaxold,partmp(10)
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /inout/xin(100,100),yin(100,100),xout(100,100),
     &yout(100,100)
      Common /parameter/parm(100,10)
      Common /state/lflast,lftin,lftout,ndpstn,iup
      Common /commands/nmax,epsiln,delin,delout
      Common /max/nptmax
      Common /screenbnd/xleft,xright,ybot,ytop
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character open*1,solid*1,filtyp*1,plyclr*1,black*1,white*1
      Character intype*1,ans*2,ans2*2
C
      itest=0
      If (ntotal.GT.0.AND.npoly.GT.0) Then
C
C - Store the old polygon information into the oldcoord common block.
C
         Call storeold(npoly,itest3)
C
C - Delete the old polygon information associated with npoly.
C
         ntotal=ntotal-1
         If (ntotal.LT.0) ntotal=0
         ngbloc=ngbloc+1
         ngbtop=ngbtop+1
         ierr3=2
         Call deltpl(npoly,ierr3)
         If (ierr3.LE.-1) Then
            Call wrtmsg('  Error in testopo routine')
            itest=-1
            Go To 100
         End If
         Call dealloc(npoly,ierr3)
C
C - Find the available polygon position npoly2.
C
         Call fndnum(npoly2)
C
C - Test polygon in xtemp,ytemp.
C
         Call fndtopnew(ntop)
         Call fndtpl(npoly2,ntop,itest2)
C
         If (itest2.EQ.1) Then
            ntotal=ntotal+1
            Call stoply(xtemp,ytemp,ntemp,npoly2,ierr3)
            Call fndbbx(xmin,xmax,ymin,ymax,xtemp,ytemp,ntemp,
     &      celout)
            xminbx(npoly2)=xmin
            xmaxbx(npoly2)=xmax
            yminbx(npoly2)=ymin
            ymaxbx(npoly2)=ymax
            Call fndbpl(npoly2,ierr3)
            Do 95 ii=1,10
               parm(npoly2,ii)=partmp(ii)
   95       Continue
C
C - Map the new topology structure onto the old topology arrays.
C
            Call newold
C
C - Undraw the old polygon and draw the new polygon.
C
            If (izoom.EQ.1) Then
               Do 50 i=1,nold
                  Call trans(xold(i),yold(i),xold(i),yold(i))
   50          Continue
               Do 55 j=1,ntemp
                  Call trans(xtemp(j),ytemp(j),xtemp(j),ytemp(j))
   55          Continue
            End If
C
C - Clip and undraw the old polygon.
```

```
C
            Call setclr(black)
            Call setfil(open)
            Call drwclp(xold,yold,nold,xleft,xright,ybot,ytop)
C
C - Clip and draw the new polygon.
C
            Call setclr(plyclr)
            Call drwclp(xtemp,ytemp,ntemp,xleft,xright,ybot,ytop)
            Call setfil(solid)
            itest=1
         Else If (itest2.EQ.-1) Then
C
C - Since new polygon failed test, restore the old polygon infor-
C   mation at the position of npoly2.
C
            ntotal=ntotal+1
            Call stoply(xold,yold,nold,npoly2,ierr2)
            xminbx(npoly2)=xminold
            xmaxbx(npoly2)=xmaxold
            yminbx(npoly2)=yminold
            ymaxbx(npoly2)-ymaxold
C
C - Restore the inner and outer bounding polygon info.
C
            Do 60 jj=1,nold
               xin(npoly2,jj)=xinold(jj)
               yin(npoly2,jj)=yinold(jj)
               xout(npoly2,jj)=xoutold(jj)
               yout(npoly2,jj)=youtold(jj)
   60       Continue
C
C - Restore the paramter info.
C
            Do 65 j=1,10
               parm(npoly2,j)=partmp(j)
   65       Continue
C
C - Map the old topology structure back onto the new topology array.
C
            info(ndpstn)=npoly2
            Call oldnew
            itest=-1
         End If
      End If
C
100   Continue
      Return
      End
C+
C **************************************************************
C  testspecs - Test the grid specifications in the two common blocks
C              gridspecs and gridspecs2 for the matching of essential
C              parameters.
C-
C **************************************************************
      Subroutine testspecs(itest)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /gridspecs2/id2,pgm2,nc2,nr2,nz2,xo2,dx2,yo2,dy2,
     &iproj2,cm2,bl2
      Character id*56,id2*56,pgm*8,pgm2*8
C
      If ((nc.EQ.nc2).AND.(nr.EQ.nr2)) Then
         itest=1
      Else
         Call wrtmsg(' Error TESTSPECS, grid specifications do not
     &   match')
         itest=-1
      End If
C
      Return
      End
C+
C *************************************************************
C  totalt - Computes the total turning angle in degrees of the
C           given polygon.  The total turning angle is the sum
C           of the individual heading changes when moving from
C           corner point to corner point around a polygon.
C-
C *************************************************************
      Subroutine totalt(gamma,xpoly,ypoly,nbrpts)
      Dimension xpoly(nbrpts),ypoly(nbrpts)
C
      gamma=0.0
      theta=0.0
C
      Do 10 i=1,nbrpts
      If (i.EQ.1) Then
         iback=nbrpts
         inext=2
      Else If (i.EQ.nbrpts) Then
         iback=nbrpts-1
         inext=1
      Else
         iback=i-1
         inext=i+1
      End If
C
      x1=xpoly(iback)
      y1=ypoly(iback)
      x2=xpoly(i)
      y2=ypoly(i)
      x3=xpoly(inext)
      y3=ypoly(inext)
C
C - Compute the next relative heading change when going from the line
C   segment (x1,y1),(x2,y2) to (x2,y2),(x3,y3).
C
      Call tangle(theta,x1,y1,x2,y2,x3,y3)
C
C - To determine if the heading change was to the left (counter-clockwise
C   or the right (clockwise), construct the components of two
C   vectors A and B, and compute their cross-product.  Since the
C   vectors are in a plane the k (or z) components of the vectors
C   will be zero.
C
      Ax=x2-x1
      Ay=y2-y1
      Bx=x3-x2
      By=y3-y2
C
C - Now compute the cross-product; since the k (or z) components of
C   the vectors A and B are zerro we need only compute the k (or z)
C   component of vector C.
C
      Cz=Ax*By-Ay*Bx
C
C - We make the following definitions:
```

```
C
C       Cz > 0.0 - Relative heading change was to the left.
C       Cz < 0.0 - Relative heading change was to the right.
C
      If (Cz.GT.0.0) Then
         parity=-1.0
      Else If (Cz.LT.0.0) Then
         parity=1.0
      Else
         parity=0.0
      End If
      gamma=gamma+parity*theta
   10 Continue
C
      Return
      End
C+
C **************************************************************
C  trans  - Transforms a coordinate (xold,yold) from the normal
C           unzoomed screen to a zoomed screen (xnew,ynew).
C-
C **************************************************************
      Subroutine trans(xnew,ynew,xold,yold)
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
C
      xorg=iwcorg+(ncmin-1)*nxorgp
      yorg=jwcorg+(nrmin-1)*nyorgp
C
      xnew=iwc0+(xold-xorg)*nxpix/nxorgp
      ynew=jwc0+(yold-yorg)*nypix/nyorgp
C
      Return
      End
C+
C **************************************************************
C  tstbnd - Test the location xtest,ytest to see if it is within the
C           bounding box around polygon npoly.
C
C           inout = Flag for point being inside or outside of box.
C                 = 1 - Point is inside bounding box
C                 = 0 - Point is outside of bounding box
C                 =-1 - npoly passed to tstbnd out of range.
C-
C **************************************************************
      Subroutine tstbnd(inout,xtest,ytest,npoly)
      Common /box/xminbx(100),xmaxbx(100),yminbx(100),ymaxbx(100)
      Common /commands/nmax,epsiln,delin,delout
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Then
         inout=0
C
         If ((xtest.LE.xmaxbx(npoly)).AND.(xtest.GE.xminbx(npoly)).AND.
     &   (ytest.LE.ymaxbx(npoly)).AND.(ytest.GE.yminbx(npoly)))inout=1
      Else
         inout=-1
      End If
C
      Return
      End
C+
C **************************************************************
C  tstend - TeST END - Tests the endpoints of the line (x1,y1),
c           (x2,y2) for mapping onto region 0000 (See "Principles
c           of Interactive Computer Graphics", by Newman and
c           Sproull, figure 5-5, page 66.)
c
c           Region 0000 is defined as any x,y:
c              xleft < x < xright
c              ybot  < y < ytop
c
c             inout = Value describing properties of pair
c                 3 = Both points in region 0000
c                 2 = point (x2,y2) within region 0000
c                 1 = point (x1,y1) within region 0000
c                 0 = line segment crosses region 0000 (endpoints
c                     are outside)
c                -1 = line segment is entirely off screen
c
c-
c*************************************************************
      Subroutine tstend(inout,x1,y1,x2,y2,xleft,xright,ybot,ytop)
c
      Call fndcde(ione4,ione3,ione2,ione1,x1,y1,xleft,xright,ybot,ytop)
      Call fndcde(itwo4,itwo3,itwo2,itwo1,x2,y2,xleft,xright,ybot,ytop)
c
      nsum1=ione4+ione3+ione2+ione1
      nsum2=itwo4+itwo3+itwo2+itwo1
c
      inout=-1
      If ((nsum1.EQ.0).AND.(nsum2.EQ.0)) Then
         inout=3
      Else If (nsum1.EQ.0) Then
         inout=1
      Else If (nsum2.EQ.0) Then
         inout=2
      Else
         icount=0
         If ((ione4.EQ.1).AND.(itwo4.EQ.1)) Then
            icount=icount+1
         End If
c
         If ((ione3.EQ.1).AND.(itwo3.EQ.1)) Then
            icount=icount+1
         End If
c
         If ((ione2.EQ.1).AND.(itwo2.EQ.1)) Then
            icount=icount+1
         End If
c
         If ((ione1.EQ.1).AND.(itwo1.EQ.1)) Then
            icount=icount+1
         End If
c
         If (icount.EQ.0) Then
            delx=x2-x1
            dely=y2-y1
            If (abs(delx).LT.1.e-16) delx=sign(1.e-16,delx)
            If (abs(dely).LT.1.e-16) dely=sign(1.e-16,dely)
c
            slope=dely/delx
            b=y2-slope*x2
c
            Do 10 i=1,2
               If (i.EQ.1) Then
                  xtest=xleft
               Else
```

```
                  xtest=xright
               End If
c
               ytest=slope*xtest+b
               If (ytest.GE.ybot.AND.ytest.LE.ytop) inout=0
10          Continue
c
            Do 20 j=1,2
               If (j.EQ.1) Then
                  ytest=ybot
               Else
                  ytest=ytop
               End If
c
               xtest=(ytest-b)/slope
               If (xtest.GE.xleft.AND.xtest.LE.xright) inout=0
20          Continue
         End If
      End If
C
      Return
      End
C+
C ************************************************************
C  tstpgn - TeST PolyGoN - Tests the two polygons npoly and
C           (xtemp,ytemp), under control of icon, for whether
C           one polygon is inside, outside, or if they cross.
C
C        icon  = Input flag controlling how polygon comparisions
C                are made.
C
C              = 1 = Tests if polygon (xtemp,ytemp) is within
C                    polygon npoly (xscr,yscr)
C
C              = 0 = Tests if polygon npoly (xscr,yscr) is with-
C                    in polygon (xtemp,ytemp)
C
C        inout = Flag describing whether a polygon is inside,
C                outside, or crosses another polygon.
C
C              = 1 = inside
C              = 0 = outside
C              =-1 = cross
C
C
C-
C ************************************************************
      Subroutine tstpgn(inout,npoly,icon,itest)
      Dimension xtest(100),ytest(100),xpoly(100),ypoly(100)
      Common /commands/nmax,epsiln,delin,delout
      Common /temp/ntemp,xtemp(100),ytemp(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Then
         itest=1
         nbrpts=numply(npoly)
         If (icon.EQ.1) Then
            Do 10 i=1,ntemp
            xtest(i)=xtemp(i)
            ytest(i)=ytemp(i)
   10       Continue
C
            Do 20 ii=1,nbrpts
            xpoly(ii)=xscr(npoly,ii)
            ypoly(ii)=yscr(npoly,ii)
```

```
 20         Continue
C
            ncount=ntemp
            npp=nbrpts
         Else If (icon.EQ.0) Then
            Do 30 j=1,nbrpts
            xtest(j)=xscr(npoly,j)
            ytest(j)=yscr(npoly,j)
 30         Continue
C
            Do 40 jj=1,ntemp
            xpoly(jj)=xtemp(jj)
            ypoly(jj)=ytemp(jj)
 40         Continue
            ncount=nbrpts
            npp=ntemp
         Else
            itest=-1
            Call wrtmsg(' Error in tstpgn, icon was not 1 or 0')
            Go To 100
         End If
         nout=0
         nin=0
C
         Do 50 k=1,ncount
         Call plytst(xpoly,ypoly,npp,xtest(k),ytest(k),iresit)
         If (iresit.EQ.0) Then
            nout=nout+1
         Else If (iresit.EQ.1) Then
            nin=nin+1
         End If
 50      Continue
C
         If (nout.EQ.ncount) Then
            inout=0
         Else If (nin.EQ.ncount) Then
            inout=1
         Else
            inout=-1
         End If
      Else
         Call wrtmsg(' Error, npoly not in the range: 1=<npoly=<nmax')
         itest=-1
      End If
C
 100  Continue
      Return
      End
C*
C*****************************************************************
C tstseg - Tests two line segments [x1,y1],[x2,y2] and [xt1,yt1],
C          [xt2,yt2] to see if they cross.
C
C          The test consists of:
C
C            (i) Comparing sides for overlap of their range
C                and domain.
C
C           (ii) If sides overlap from (i), calculate the
C                intersection point (xint,yint).
C
C          (iii) Determine if the intersection point lies in
C                the overlap of the sides' ranges and domains
C
```

```
        ncross = Flag telling if sides cross
               = 1 = sides cross
               = 0 = sides do not cross

    Author: Bruce A. Chuchel, USGS, Menlo Park, CA., 1/85
-
***************************************************************
    Subroutine tstseg(ncross,x1,y1,x2,y2,xt1,yt1,xt2,yt2)

    ncross=0

- For the side joining the points [x1,y1] and [x2,y2] construct
  the domain and range intervals [xlow,xup] and [ylow,yup]

    xlow=amin1(x1,x2)
    ylow=amin1(y1,y2)
    xup=amax1(x1,x2)
    yup=amax1(y1,y2)

    txlow=amin1(xt1,xt2)
    tylow=amin1(yt1,yt2)
    txup=amax1(xt1,xt2)
    tyup=amax1(yt1,yt2)

- Test (i)

    If ((txlow.GE.xlow.OR.txup.GE.xlow).AND.
   &(txlow.LE.xup.OR.txup.LE.xup).AND.
   &(tylow.GE.ylow.OR.tyup.GE.ylow).AND.
   &(tylow.LE.yup.OR.tyup.LE.yup)) Then

- Determine the slope (dely/delx), and y-axis intercept (b),
  of the line segment between the points [x1,y1] and [x2,y2].

       delx=x2-x1
       dely=y2-y1
       If (abs(delx).LT.1.0e-16)delx=sign(1.0e-16,delx)
       If (abs(dely).LT.1.0e-16)dely=sign(1.0e-16,dely)
       b=y1-x1*(dely/delx)

- Test (ii)

       tdelx=(xt2-xt1)
       tdely=(yt2-yt1)
       If (abs(tdelx).LT.1.0e-16)tdelx=sign(1.0e-16,tdelx)
       If (abs(tdely).LT.1.0e-16)tdely=sign(1.0e-16,tdely)
       tb=yt1-xt1*(tdely/tdelx)

       diff=((dely/delx)-(tdely/tdelx))
       If (abs(diff).LT.1.0e-16)diff=sign(1.0e-16,diff)
       xint=(tb-b)/diff
       yint=xint*(dely/delx)+b

- Test (iii)
  Now test (xint,yint) for being in the intervals bounded
  by [xlow,xup],[ylow,yup] and [txlow,txup],[tylow,tyup]

       If ((xint.GE.xlow.AND.xint.LE.xup).AND.
   &   (yint.GE.ylow.AND.yint.LE.yup).AND.
   &   (xint.GE.txlow.AND.xint.LE.txup).AND.
   &   (yint.GE.tylow.AND.yint.LE.tyup)) Then
          ncross=1
       End If
    End If
```

```
      Return
      End
C*
C*****************************************************************
C tstxcr - Tests whether the polygon given in xtemp,ytemp array
C          overlaps on the polygon npoly.
C
C          The test consists of:
C
C            (i) Comparing sides in pairs for overlap of their
C                range and domain.
C
C           (ii) If sides overlap, calculate the intersection
C                point (xint,yint).
C
C          (iii) Determine if the intersectior point lies in
C                the overlap of the sides' ranges and domains
C
C
C          ncross = Flag telling if polygons overlap
C                 = 1 = polygons overlap
C                 = 0 = polygons do not overlap
C
C          npoly  = Pointer to polygon in xscr,yscr arrays
C-
C*****************************************************************
      Subroutine tstxcr(ncross,npoly)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /temp/ntemp,xtemp(100),ytemp(100)
C
      itest=0
      ncross=0
      nbrpts=numply(npoly)
C
C - Start the counter (i) for the side of the polygon npoly.
C
      i=1
      Do 10 while(itest.EQ.0.AND.i.GE.1.AND.i.LE.nbrpts)
      If (i.EQ.nbrpts) Then
         x1=xscr(npoly,nbrpts)
         y1=yscr(npoly,nbrpts)
         x2=xscr(npoly,1)
         y2=yscr(npoly,1)
      Else
         x1=xscr(npoly,i)
         y1=yscr(npoly,i)
         x2=xscr(npoly,i+1)
         y2=yscr(npoly,i+1)
      End If
C
C - For the side of the polygon joining the points (x1,y1) and
C   (x2,y2) construct the domain and range intervals [xlow,xup],
C   [ylow,yup].
C
      xlow=amin1(x1,x2)
      ylow=amin1(y1,y2)
      xup=amax1(x1,x2)
      yup=amax1(y1,y2)
C
C - Determine the slope (dely/delx), and y-axis intercept (b),
C   of the line segment between the points (x1,y1) and (x2,y2).
C
      delx=x2-x1
      dely=y2-y1
      If (abs(delx).LT.1.0e-12)delx=sign(1.0e-12,delx)
      If (abs(dely).LT.1.0e-12)dely=sign(1.0e-12,dely)
      b=y1-x1*(dely/delx)
C
C - Now start searching through the xtemp,ytemp array and do the
C   tests outlined above.  Start the counter (icnt) for the side
C   of polygon in xtemp,ytemp.
C
      icnt=1
      Do 20 while(itest.EQ.0.AND.icnt.GE.1.AND.icnt.LE.ntemp)
      If (icnt.EQ.ntemp) Then
         xt1=xtemp(ntemp)
         yt1=ytemp(ntemp)
         xt2=xtemp(1)
         yt2=ytemp(1)
      Else
         xt1=xtemp(icnt)
         yt1=ytemp(icnt)
         xt2=xtemp(icnt+1)
         yt2=ytemp(icnt+1)
      End If
C
      txlow=amin1(xt1,xt2)
      tylow=amin1(yt1,yt2)
      txup=amax1(xt1,xt2)
      tyup=amax1(yt1,yt2)
C
C - Test (i)
C
      If ((txlow.LT.xlow.AND.txup.LT.xlow).OR.
     &(txlow.GT.xup.AND.txup.GT.xup).OR.
     &(tylow.LT.ylow.AND.tyup.LT.ylow).OR.
     &(tylow.GT.yup.AND.tyup.GT.yup)) Then
         itest=0
      Else
C
C - Test (ii)
C
         tdelx=(xt2-xt1)
         tdely=(yt2-yt1)
         If (abs(tdelx).LT.1.0e-12)tdelx=sign(1.0e-12,tdelx)
         If (abs(tdely).LT.1.0e-12)tdely=sign(1.0e-12,tdely)
         tb=yt1-xt1*(tdely/tdelx)
C
         diff=((dely/delx)-(tdely/tdelx))
         If (abs(diff).LT.1.0e-12)diff=sign(1.0e-12,diff)
         xint=(tb-b)/diff
         yint=xint*(dely/delx)+b
C
C - Test (iii)
C   Now test (xint,yint) for being in the intervals bounded
C   by [xlow,xup],[ylow,yup] and [txlow,txup],[tylow,tyup]
C
         If ((xint.GT.xlow.AND.xint.LT.xup).AND.
     &   (yint.GT.ylow.AND.yint.LT.yup).AND.
     &   (xint.GT.txlow.AND.xint.LT.txup).AND.
     &   (yint.GT.tylow.AND.yint.LT.tyup)) Then
            ncross=1
            itest=1
         End If
      End If
      If (ncross.EQ.0)icnt=icnt+1
   20 End Do
```

```
      If (itest.EQ.0)i=i+1
   10 End Do
C
      Return
      End
C+
C ***************************************************************
C  unzcom - Driver to redraw unzoomed grid on Envision terminal.
C-
C ***************************************************************
      Subroutine unzcom(itest)
      Common /misc/ncol,nrow,first,ntop,ifirst
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Common /model/mdflag
      Character plyclr*1,black*1,white*1,open*1,solid*1,filtyp*1
      Character first*1
C
      itest=1
      itest2=1
C
      If (ifirst.EQ.1) Then
         Call intscr
         Call clrsgd(1,itest2)
         If (itest2.GE.0) Then
            Call clrply
            ifirst=0
            iwcorg=iwc0
            jwcorg=jwc0
            nxorgp=nxpix
            nyorgp=nypix
         End If
      Else
         Call intscr
         Call clrsgd(0,itest2)
         Call clrply
      End If
      If (itest2.GE.0) Then
         izoom=0
         xstart=ncmin
         ystart=nrmin
         xsc=float(nxpix)
         ysc=float(nypix)
         Call setbnd
      End If
      If (itest2.GE.0.AND.mdflag.EQ.1) Then
         Call setclr(plyclr)
         Call setfil(open)
         Call drawwalk
         Call setfil(solid)
      End If
      If (itest2.LT.0)itest=itest2
C
      Return
      End
C+
C     UPSHIFT - Converts lowercase to uppercase.
C               All non-lowercase characters unchanged.
```

```
C
C     CALL UPSHIFT(A)
C
C             A = A character string of any length
C
C       Author: Richard W. Saltus, USGS, Denver, CO.
C-
      SUBROUTINE UPSHIFT(A)
      CHARACTER*(*) A,UP*(26),DOWN*(26)
      UP='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
      DOWN='abcdefghijklmnopqrstuvwxyz'
      ILEN=LEN(A)
      DO 10 I=1,ILEN
        INUM=INDEX(DOWN,A(I:I))
        IF (INUM.NE.0) A(I:I)=UP(INUM:INUM)
  10      CONTINUE
      RETURN
      END
C+
C ************************************************************
C  valchg - Allows the changing of the parameterss associated with
C           polygon npoly.
C
C           npoly = Polygon number (1-nmax)
C
C           iset  = Sets which arrays in parm* arrays are to
C                   be reset.
C                 = 0 = All parameters (1-10) will be prompted for
C                 = 1 - 10 = Prompts only for the one parameter
C                         selected
C
C-
C ************************************************************
      Subroutine valchg(npoly,iset,itest)
      Common /commands/nmax,epsiln,delin,delout
      Common /labels/label
      Common /parameter /parm(100,10)
      Character label(10)*15,ulabel*15,quest*80
C
      If (npoly.GE.1.AND.npoly.LE.nmax) Then
         itest=0
         If (iset.EQ.0) Then
            ievent=0
            icnt=1
         Else If (iset.GE.1.AND.iset.LE.10) Then
            ievent=1
            icnt=iset
         Else
            Print *,' Error, iset out of range'
            itest=-1
         End If
C
C - Print quit message
C
         Call wrtmsg(' Enter // when done assiging parameters to
     &   thispolygon')

C
         Do 10 while(itest.EQ.0)
            plyval=parm(npoly,icnt)
            ulabel=label(icnt)
C
            quest=ulabel//' parameter'
            ival=irquest(quest,plyval,'(e15.8)',15)
```

```
C
            If (ival.EQ.1) Then
               perm(npoly,icnt)=plyval
            Else If (ival.EQ.-1) Then
               itest=-1
            End If
            icnt=icnt+1
            If (ievent.EQ.1.OR.icnt.GT.10.AND.itest.NE.-1)itest=1
   10    End Do
      Else
         Call wrtmsg(' Error, npoly out of range')
         itest=-1
      End If
C
      Return
      End
C+
C ************************************************************
C  wait  - Waits for user to enter a carriage return before
C          continuing.
C-
C ************************************************************
      Subroutine wait
      Character ans*1
C
      Write (6,10)
   10 Format (/,' Enter a carriage return <CR> to continue:',$)
      Read (5,20)ans
   20 Format (a1)
C
      Return
      End
C+
C ************************************************************
C  walk - Starting at the position of polygon NGON in the tree
C         (topology array), WALK returns the node position of the
c         next polygon in the tree.  The walk proceeds as follows:
C
C           (1) Move down the (sub)tree on the 'left' side, by
C               looking at idown of the current position.  If this
C               is not zero return this as the next polygon.
C
C           (2) When a polygon is not found 'below' the current
C               position (idown is zero), look to the right of
C               this position and test:
C
C               (a) If a polygon is found on the right, return
C                   this polygon.
C               (b) If a polygon is not found on the right, move
C                   up a level and reapply test (a).
C
C           (3) When no more polygons are found to the right and
C               above the current position, the tree has been
C               completely searched.
C
C   Variables:
C
C         next = Next polygon in tree
C                >0 = a polygon exists
C                 0 = tree has been exhausted
C
C         ngon = Position in tree where walk is to start
C                (ngon must be greater than zero).
C
```

```
C        itest = Error flag
C                1 = Next polygon was found
C                0 = No more polygons
C               -1 = A value in tree is less than 0.
C
C       Note: ngon and next are pointers to positions in the
C             topology arrays.
C
C
C-
C ***************************************************************
      Subroutine walk(next,ngon,itest)
      Common /topology/ info(100),iupper(100),idown(100),
     & ileft(100),iright(100)
      Common /commands/ nmax,epsiln,delin,delout
C
      next=0
      If (ngon.ge.1.and.ngon.le.nmax) then
         npnt=ngon
         next=idown(ngon)
         If (next.GT.0) then
            itest=1
         Else If (next.EQ.0) Then
            next=iright(ngon)
            If (next.GT.0) Then
               itest=1
            Else If (next.EQ.0) Then
               itest2=0
               i=1
               Do 20 while(itest2.EQ.0)
                  npnt=iupper(npnt)
                  If (npnt.GT.0) Then
                     next=iright(npnt)
                     If (next.GT.0) Then
                        itest=1
                        itest2=1
                     Else If (next.EQ.0) Then
                        itest2=0
                     Else
                        itest2=-1
                        itest=-1
              print *,' Error, iright was out of range (iright<=0)'
                     End If
                  Else If (npnt.EQ.0) Then
                     next=0
                     itest=0
                     itest2=1
                  Else
                     itest=-1
                     itest2=-1
                     print *,' Error, iup of ',npnt,' less than zero'
                  End If
                  i=i+1
                  If (i.GT.nmax) Then
                     print *,' Error, i (counter) was >',nmax
                     itest=-1
                     itest2=-1
                  End If
20             End Do
            Else
               print *,' Error, iright of ',npnt,' was less than zero'
               itest=-1
            End If
         Else
            print *,' Error, idown of ',ngon,' was less than zero'
            itest=-1
         End If
      Else
         print *,' Error, NGON passed to WALK out of range'
         itest=-1
      End if
C
      Return
      End
C+
C ************************************************************
C  wcbp - World coordinate byte packing.  Converts a world co-
C         ordinate pair into the (hexadecimal) code required for
C         the Envision terminal.  See wcbp.inf and Envision manual.
C
C         Author: Robert Simpson, USGS, Menlo Park, CA, 10/83.
C-
C ************************************************************
      Character*5 Function wcbp(i,j)
      Character blank*1
      Parameter (imax=16284,jmax=16284)
      Parameter (imin=0,jmin=0)
      Parameter (blank=' ',iblank=ichar(' '))
C
C - Force i,j into bounds...
      iin=min(imax,max(imin,i))
      jin=min(jmax,max(jmin,j))
C - Get hi and lo bytes and offset with blank...
      ilo=mod(iin,64)+iblank
      imed=iin/64+iblank
      jlo=mod(jin,64)+iblank
      jmed=jin/64+iblank
C - Put bytes together...
      wcbp=blank//char(imed)//char(ilo)//char(jmed)//char(jlo)
C
      Return
      End
C+
C ************************************************************
C  wrhead - Writes the header for a standard grid (new version).
C
C        Author: Robert W. Simpson, USGS, Menlo Park, CA.
C
C-
C ************************************************************
      Subroutine wrhead(unit,id,pgm,ncol,nrow,nz,
     &xo,dx,yo,dy,iproj,cm,bl,itest)
      Character id*56,pgm*8
      Integer unit
C
      itest=1
      Write (unit)id,pgm,ncol,nrow,nz,xo,dx,yo,dy,iproj,cm,bl
C
      Return
      End
C+
C ************************************************************
C  wrtgrd - WRITe GRID - Writes out wrkgrd as a Denver standard
C           grid file to 'name'.
C-
C ************************************************************
      Subroutine wrtgrd(name,itest)
      Common /work/wrkgrd(250000)
      Common /gridspecs2/id2,pgm2,nc2,nr2,nz2,xo2,dx2,yo2,dy2,
     &iproj2,cm2,bl2
      Character name*80,id2*56,pgm2*8
C
      itest=1
      pgm2='polygon '
      dummy=0.0
      Open (11,file=name,status='new',form='unformatted')
      Call wrhead(11,id2,pgm2,nc2,nr2,nz2,xo2,dx2,yo2,dy2,iproj2,
     &cm2,bl2,itest)
      If (itest.GE.0) Then
         Do 20 j=1,nr2
         Write (11)dummy,(wrkgrd(i+(j-1)*nc2),i=1,nc2)
   20    Continue
      End If
      Close (11)
C
  100 Continue
      Return
      End
C+
C **************************************************************
C  wrtmod - Writes out a Polyogn model file.
C
C
C-
C **************************************************************
      Subroutine wrtmod(itest)
      Common /topology/info(100),iupper(100),idown(100),
     &ileft(100),iright(100)
      Common /screenloc/ntotal,numply(100),xscr(100,100),yscr(100,100)
      Common /parameter /parm(100,10)
      Common /labels/label
      Common /commands/nmax,epsiln,delin,delout
      Common /junk/ngbtop,jnktop(100),ngbloc,jnkloc(100)
      Common /names/grdnam,modnam,modgrd
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
C
      Character label(10)*15
      Character id*56,pgm*8,grdnam*80,modnam*80,modgrd*80
C
      Open (10,file=modnam,status='unknown',form='formatted',
     &carriagecontrol='list')
C
C - Write out the current parms assigned to standard grid header
C   information.
C
C     write (10,5) id,pgm,nc,nr,nz
C5     format(x,a56,x,a8,3(x,i3))
C     write (10,10) xo,dx,yo,dy
C10    format(4(x,e15.8))
C     write (10,15) iproj,cm,bl
C15    format(x,i3,2(x,e15.8))
C
C - Write out the number of polygons in the model and the
C   labels assigned to the parm arrays.
C
      Write (10,20)ntotal
   20 Format (x,i3)
      Write (10,25)(label(k),k=1,5)
      Write (10,25)(label(k),k=6,10)
   25 Format (5(x,a15))
C
C - Write out the polygon number, the number of points in this
```

```
C   polygon, the grid locations and the polygon parms.
C
      delxcn=dx/nxorgp
      delycn=dy/nyorgp
      Do 60 i=1,nmax
      npoly=info(i)
      If (npoly.GT.0) Then
         nbrpts=numply(npoly)
         Write (10,33)npoly,nbrpts
   33    Format (2(x,i3))
         Write (10,35)(parm(npoly,kk),kk=1,5)
         Write (10,35)(parm(npoly,kk),kk=6,10)
   35    Format (5(x,e15.8))
         Do 40 j=1,nbrpts
         xgrid=xo+((xscr(npoly,j)-iwcorg)*delxcn)
         ygrid=yo+((yscr(npoly,j)-jwcorg)*delycn)
         Write (10,45)xgrid,ygrid
   45    Format (2(x,e16.8))
   40    Continue
      End If
   60 Continue
C
C - Write out the topology structure
C
      Do 70 ndpstn=1,nmax
      Write (10,65)info(ndpstn),iupper(ndpstn),idown(ndpstn),
     &ileft(ndpstn),iright(ndpstn)
   65 Format (5(x,i3))
   70 Continue
C
      Close (10)
      itest=1
      Return
      End
C+
C *************************************************************
C  wrtmsg - Writes out a text string to the terminal.
C-
C *************************************************************
      Subroutine wrtmsg(text)
      Character text*(*)
C
      ileng=itlen(text)
      Write (6,10)text
   10 Format (x,a<ileng>)
C
      Return
      End
C+
C *************************************************************
C  zomcom - Zoom command mode.  Controls the function of zooming
C           and unzoomining in the grid.
C
C     Options:
C
C           c = Clear zoom stack
C           d = Draw the grid using a zoom value selected from the
C               stack.
C           r = Recall and draw the grid using the zoom values
C               currently pointed to in stack by nzoom.
C           s = Select a subgrid (using mouse, cursor, grid
C               coords.)
C           u = Unzoom grid, draw full grid on screen (does nothing
C               to zoom values)
```

```
C           h = Help message
C           q = Quit and return to Polygon command level
C
C
C-
C **************************************************************
      Subroutine zcmcom(ifirst,itest)
      Character ans*2,quest*80,zcom*2,first*1
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /scale/xsc,ysc,xstart,ystart,xinit,yinit
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /scalefacts/iwc0,jwc0,nxpix,nypix,pixdim
      Common /original/iwcorg,jwcorg,nxorgp,nyorgp
      Common /flags/mcflag,votflg
      Common /model/mdflag
      Common /colors/plyclr,black,white
      Common /fill/open,solid,filtyp
      Character plyclr*1,black*1,white*1,open*1,solid*1,filtyp*1
      Character mcflag*2,votflg*2
C
      Call enhmsg('*** Zoom mode ***')
C
C - Testing of input variables goes here
C
      zcom='H'
      iflag=0
      Do 10 while(iflag.EQ.0)
      quest=' Zoom command (c/d/r/s/u/h/q)'
      ival=iaquest(quest,zcom,'(a2)',-2)
C
      If (zcom.EQ.'C') Then
         Call askans(ans)
         If (ans.EQ.'Y') Then
            Call intzom
            Call setbnd
         Else If (ans.EQ.'Q') Then
            iflag=-1
         End If
      Else If (zcom.EQ.'D') Then
         If (nzoom.GE.1) Then
            Call zompck(itest)
            If (itest.GE.0) Then
               Call displa
               Call scaleg2sc(0)
               Call clrsgd(0,itest)
               xstart=ncmin
               ystart=nrmin
               Call setbnd
            End If
            If (mdflag.EQ.1.AND.itest.GE.0) Then
               Call setclr(plyclr)
               Call setfil(open)
               Call drawwalk
               Call setfil(solid)
            End If
         Else
            Write (6,50)
   50       Format (/,' Sorry, zoom stack is empty...',/)
         End If
      Else If (zcom.EQ.'S') Then
         If (ifirst.EQ.1) Then
            Call intscr
            Call clrsgd(1,itest)
            Call clrply
         xsc=float(nxpix)
         ysc=float(nypix)
         iwcorg=iwc0
         jwcorg=jwc0
         nxorgp=nxpix
         nyorgp=nypix
         Call setbnd
      End If
      If (mdflag.EQ.1.AND.ifirst.EQ.1) Then
         Call setclr(plyclr)
         Call setfil(open)
         Call drawwalk
         Call setfil(solid)
      End If
      ifirst=0
20    Continue
      Call asktyp(itype,itest)
      If (itest.EQ.-1)Go To 30
      If (nzoom.EQ.5)Call zomstk(itest)
      If (itest.EQ.-1)Go To 30
      If (itype.EQ.1) Then
         Call getsub(itest)
      Else If (itype.EQ.2) Then
         Call asksub(itest)
      End If
      If (itest.EQ.1)izoom=1
      If (itest.EQ.-1)Go To 20
30    Continue
      If (itest.GE.0) Then
         Call displa
         Call scaleg2sc(0)
         Call clrsgd(0,itest)
         xstart=ncmin
         ystart=nrmin
         Call setbnd
      End If
      If (mdflag.EQ.1.AND.itest.GE.0) Then
         Call setclr(plyclr)
         Call setfil(open)
         Call drawwalk
         Call setfil(solid)
      End If
   Else If (zcom.EQ.'R') Then
      If (nzoom.GE.1) Then
         Call zomrcl(itest)
         If (itest.GE.0) Then
            Call displa
            Call scaleg2sc(0)
            Call clrsgd(0,itest)
            Call clrply
            xstart=ncmin
            ystart=nrmin
            Call setbnd
         End If
         If (mdflag.EQ.1.AND.itest.GE.0) Then
            Call setclr(plyclr)
            Call setfil(open)
            Call drawwalk
            Call setfil(solid)
         End If
      Else
         Write (6,50)
      End If
   Else If (zcom.EQ.'U') Then
```

```
         Call unzcom(itest)
      Else If (zcom.EQ.'H') Then
         Call hlpzom
      Else If (zcom.EQ.'Q'.OR.ival.EQ.-1) Then
         iflag=-1
      Else
         Call errmsg
      End If
      zcom='Q'
   10 End Do
C
      If (iflag.EQ.-1)itest=-1
      Return
      End
C+
C *************************************************************
C  zompck - Allows the picking of a selected zoom value from the
C           zoom stack (if any exist).
C-
C *************************************************************
      Subroutine zompck(itest)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Character quest*80
C
C     call hlpzpk
C
      itest=0
      Do 10 while(itest.EQ.0)
         quest=' Stack position of zoom value (1-5, 0 or // to quit)'
         ival=iiquest(quest,izval,'(i2)',0)
C
         If (izval.GE.1.AND.izval.LE.5) Then
            If (ncminz(izval).EQ.0.AND.ncmaxz(izval).EQ.0.AND.
     &      nrminz(izval).EQ.0.AND.nrmaxz(izval).EQ.0) Then
               Write (6,15)
   15    Format (/,' Error, zoom values for this position not set',/)
            Else
               ncmin=ncminz(izval)
               ncmax=ncmaxz(izval)
               nrmin=nrminz(izval)
               nrmax=nrmaxz(izval)
               itest=1
            End If
         Else If (izval.EQ.0.OR.ival.EQ.-1) Then
            itest=-1
         Else
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C ************************************************************
C  zomrcl - Recalls the subgrid values from position nzoom
C           in the zoom stack.
C-
C ************************************************************
      Subroutine zomrcl(itest)
      Common /gridspecs/id,pgm,nc,nr,nz,xo,dx,yo,dy,iproj,cm,bl
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Character id*56,pgm*8
C
      If (nzoom.GE.1.AND.nzoom.LE.5) Then
         ncmin=ncminz(nzoom)
         ncmax=ncmaxz(nzoom)
         nrmin=nrminz(nzoom)
         nrmax=nrmaxz(nzoom)
         itest=1
      Else If (nzoom.EQ.0) Then
         itest=0
      Else
         itest=-1
      End If
C
      Return
      End
C+
C ***************************************************************
C  zomstk - Controls the release of the zoom stack.  Called by
C           subroutine zomcom (ZOOM COMMAND) if nzoom equals five.
C
C           b = Bottom of stack (oldest zoom values)
C           c = Clear stack (zeroes out zoom stack)
C           t = Top of stack (youngest zoom values)
C           h = Help message
C           q = Quit and return to ZOOM COMMAND level.
C
C           nzoom = Pointer to current position in zoom stack
C                 = 0 = if zoom stack is empty.
C                       otherwise ranges between 1 and 5.
C-
C ***************************************************************
      Subroutine zomstk(itest)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Character quest*80,ans*2
C
      itest=0
      Call msgstk
      Call hlpstk
      ans='H'
      Do 10 while(itest.EQ.0)
         quest=' What method of stack release (b/c/t/h/q)'
         ival=iaquest(quest,ans,'(a2)',0)
C
         If (ans.EQ.'B') Then
            itest=1
            Do 20 i=1,4
            ncminz(i)=ncminz(i+1)
            ncmaxz(i)=ncmaxz(i+1)
            nrminz(i)=nrminz(i+1)
            nrmaxz(i)=nrmaxz(i+1)
   20       Continue
            ncminz(5)=0
            ncmaxz(5)=0
            nrminz(5)=0
            nrmaxz(5)=0
            nzoom=4
         Else If (ans.EQ.'T') Then
            itest=1
            ncminz(5)=0
            ncmaxz(5)=0
            nrminz(5)=0
            nrmaxz(5)=0
            nzoom=4
         Else If (ans.EQ.'C') Then
            Call intzom
            itest=1
         Else If (ans.EQ.'Q'.OR.ival.EQ.-1) Then
            itest=-1
         Else If (ans.EQ.'H') Then
            Call hlpstk
         Else
            Call errmsg
         End If
   10 End Do
C
      Return
      End
C+
C ****************************************************************
C  zomstr - Stores the values of ncmin,ncmax,nrmin,nrmax, used for
C           zooming, in the zoom stack at position nzoom.
C
C           See intzom.for for a description of the variables in
C           the zoom common block.
C
C-
C ****************************************************************
      Subroutine zomstr(itest)
      Common /zoom/izoom,izval,nzoom,ncminz(5),ncmaxz(5),nrminz(5),
     &nrmaxz(5)
      Common /subgrid/icmin,icmax,irmin,irmax,ncmin,ncmax,nrmin,nrmax
C
      If (nzoom.GE.0.AND.nzoom.LT.5) Then
         itest=1
         nzoom=nzoom+1
         ncminz(nzoom)=ncmin
         ncmaxz(nzoom)=ncmax
         nrminz(nzoom)=nrmin
         nrmaxz(nzoom)=nrmax
      Else If (nzoom.GE.5) Then
         Call wrtmsg(' Zoom stack is full')
         itest=-1
      End If
C
      Return
      End
```